L Number	Hits	Search Text	DB	Time stamp
1	864	Glaser.inv.	USPAT	2004/02/25 12:01
2	11	realnetwork$1.asn.	USPAT	2004/02/25 13:36
3	3051	microsoft.asn.	USPAT	2004/02/25 12:03
4	3	microsoft.asn. and (windows near3 (media adj player$1))	USPAT	2004/02/25 12:04
5	106	mbone	USPAT	2004/02/25 12:04
6	4	mbone and microsoft.asn.	USPAT	2004/02/25 12:12
7	4	aol.asn.	USPAT	2004/02/25 12:12
9	1	american adj online.asn.	USPAT	2004/02/25 12:13
8	57	america adj online.asn.	USPAT	2004/02/25 12:32
10	1467034	(stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) same (information or content or data or information or media or audio or video)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 12:52
11	1467073	(stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) same (information or content or data or information or media or audio or video or metadata or (meta adj data))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:14
12	292698	(url$1 or address$2 or identifier$1 or uri$1 or id$1) near8 (memory or storage or buffer$1 or cache$1 or database or repository)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 12:55
13	7	realnetwork$1.asn. and server$1	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 12:55
14	5838	709/231.ccls. or 709/217-219.ccls.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 12:59
15	2343	709/212-214.ccls. or 711/118.ccls.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:01
16	1549	((725/41) or (725/46) or (725/89) or (725/92) or (725/105) or (725/1) or (725/2) or (725/87)).CCLS.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:03

17	65	(("5877755") or ("5297249") or ("5289545") or ("5283819") or ("5282028") or ("5262875") or ("5253341") or ("5247347") or ("5237322") or ("5195092") or ("5164839") or ("5132992") or ("5109482") or ("5057932") or ("5051822") or ("5001580") or ("4999806") or ("4987529") or ("4975691") or ("4963995") or ("4941123") or ("4905094") or ("4899299") or ("4845756") or ("4827256") or ("4658093") or ("4611277") or ("4581484") or ("4506387") or ("4504705") or ("4253157") or ("3882538")).PN.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:10
18	522583	host$1 or server$1	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:11
19	11463	(host$1 or server$1) and (((stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) same (information or content or data or information or media or audio or video or metadata or (meta adj data))) same ((url$1 or address$2 or identifier$1 or uri$1 or id$1) near8 (memory or storage or buffer$1 or cache$1 or database or repository)))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:13

20	459	("5956729" "6185602" "5867230" "6112226" "6119154" "6138147" "6014689" "6564248" "5761673" "6463440" "6643621" "6079566" "6212535" "6212535" "6157771" "6201925" "5751280" "5784527" "5973679" "6078664" "6128435" "5630005" "5801781" "5838678" "6018351" "6188703" "6233017" "6272658" "6285825" "6393578" "6430530" "5774666" "6122436" "6118790" "5881245" "6065050" "6076734" "6164541" "5367621" "5828370" "5835495" "5892535" "5903264" "5935240" "5945986" "5949955" "6044089" "6052148" "6134201" "6215745").pn. ("6215745" "6272190" "6356178" "6404898" "6445738" "6529146" "6396500" "5937164" "5892825" "6078908" "5742730" "5898119" "5900908" "5980262" "6157770" "6185306" "4811325" "4937807" "5189630" "5208421" "5274779" "5467274" "5631693" "5680558" "5689134" "5696500"	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:13

21	781910	(stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:36
22	4157	((stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data))) and (709/231.ccls. or 709/217-219.ccls.)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:15
23	5	(((stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data))) and (709/231.ccls. or 709/217-219.ccls.)) and (america adj online.asn.)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:15
24	94298	(stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data)) same (interleav$3 or combin$3 or synchroniz$8)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:47
25	940	((stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data)) same (interleav$3 or combin$3 or synchroniz$8)) and (((stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data))) and (709/231.ccls. or 709/217-219.ccls.))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:16

26	837	(((stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data)) same (interleav$3 or combin$3 or synchroniz$8)) and (((stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data))) and (709/231.ccls. or 709/217-219.ccls.))) and (host$1 or server$1)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:16
27	33	((((stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data)) same (interleav$3 or combin$3 or synchroniz$8)) and (((stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data))) and (709/231.ccls. or 709/217-219.ccls.))) and (host$1 or server$1)) and microsoft.asn.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:25
28	0	(((((stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data)) same (interleav$3 or combin$3 or synchroniz$8)) and (((stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data))) and (709/231.ccls. or 709/217-219.ccls.))) and (host$1 or server$1)) and microsoft.asn.) and @ad<19941130	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:49

29	25	(("5956729" "6185602" "5867230" "6112226" "6119154" "6138147" "6014689" "6564248" "5761673" "6463440" "6643621" "6079566" "6212535" "6212535" "6157771" "6201925" "5751280" "5784527" "5973679" "6078664" "6128435" "5630005" "5801781" "5838678" "6018351" "6188703" "6233017" "6272658" "6285825" "6393578" "6430530" "5774666" "6122436" "6118790" "5881245" "6065050" "6076734" "6164541" "5367621" "5828370" "5835495" "5892535" "5903264" "5935240" "5945986" "5949955" "6044089" "6052148" "6134201" "6215745").pn. ("6215745" "6272190" "6356178" "6404898" "6445738" "6529146" "6396500" "5937164" "5892825" "6078908" "5742730" "5898119" "5900908" "5980262" "6157770" "6185306" "4811325" "4937807" "5189630" "5208421" "5274779" "5467274" "5631693" "5680558" "5690113" "5696500"	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:27

30	1	((("5877755") or ("5297249") or ("5289545") or ("5283819") or ("5282028") or ("5262875") or ("5253341") or ("5247347") or ("5237322") or ("5195092") or ("5164839") or ("5132992") or ("5109482") or ("5057932") or ("5051822") or ("5001580") or ("4999806") or ("4987529") or ("4975691") or ("4963995") or ("4941123") or ("4905094") or ("4899299") or ("4845756") or ("4827256") or ("4658093") or ("4611277") or ("4581484") or ("4506387") or ("4504705") or ("4253157") or ("3882538")).PN.) and ((((stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data)) same (interleav$3 or combin$3 or synchroniz$8)) and (((stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data))) and (709/231.ccls. or 709/217-219.ccls.))) and (host$1 or server$1))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:29
31	16	((("5877755") or ("5297249") or ("5289545") or ("5283819") or ("5282028") or ("5262875") or ("5253341") or ("5247347") or ("5237322") or ("5195092") or ("5164839") or ("5132992") or ("5109482") or ("5057932") or ("5051822") or ("5001580") or ("4999806") or ("4987529") or ("4975691") or ("4963995") or ("4941123") or ("4905094") or ("4899299") or ("4845756") or ("4827256") or ("4658093") or ("4611277") or ("4581484") or ("4506387") or ("4504705") or ("4253157") or ("3882538")).PN.) and ((url$1 or address$2 or identifier$1 or uri$1 or id$1) near8 (memory or storage or buffer$1 or cache$1 or database or repository))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:34
32	110234	(access$3 or select$4 or choos$3) near5(stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:40
33	7	("5793980" | "5822524" | "5933603" | "5956321" | "5991836" | "5996015" | "6040866").PN.	USPAT	2004/02/25 13:39
34	118357	(access$3 or select$4 or choos$3 or request$3) near5(stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:41

35	119217	(access$3 or select$4 or choos$3 or request$3) near5(stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data) or file$1)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:41
36	14	5822524.URPN.	USPAT	2004/02/25 13:42
37	1117	((access$3 or select$4 or choos$3 or request$3) near5(stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data) or file$1)) and (709/231.ccls. or 709/217-219.ccls.)	USPAT	2004/02/25 13:42
38	3066	((access$3 or select$4 or choos$3 or request$3) near5(stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data) or file$1)) same ((url$1 or address$2 or identifier$1 or uri$1 or id$1) near8 (memory or storage or buffer$1 or cache$1 or database or repository))	USPAT	2004/02/25 13:43
39	2	(((access$3 or select$4 or choos$3 or request$3) near5(stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data) or file$1)) same ((url$1 or address$2 or identifier$1 or uri$1 or id$1) near8 (memory or storage or buffer$1 or cache$1 or database or repository))) and ((("5877755") or ("5297249") or ("5289545") or ("5283819") or ("5282028") or ("5262875") or ("5253341") or ("5247347") or ("5237322") or ("5195092") or ("5164839") or ("5132992") or ("5109482") or ("5057932") or ("5051822") or ("5001580") or ("4999806") or ("4987529") or ("4975691") or ("4963995") or ("4941123") or ("4905094") or ("4899299") or ("4845756") or ("4827256") or ("4658093") or ("4611277") or ("4581484") or ("4506387") or ("4504705") or ("4253157") or ("3882538")).PN.)	USPAT	2004/02/25 13:44

40	8	(((access$3 or select$4 or choos$3 or request$3) near5(stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data) or file$1)) same ((url$1 or address$2 or identifier$1 or uri$1 or id$1) near8 (memory or storage or buffer$1 or cache$1 or database or repository))) and (("5956729" "6185602" "5867230" "6112226" "6119154" "6138147" "6014689" "6564248" "5761673" "6463440" "6643621" "6079566" "6212535" "6212535" "6157771" "6201925" "5751280" "5784527" "5973679" "6078664" "6128435" "5630005" "5801781" "5838678" "6018351" "6188703" "6233017" "6272658" "6285825" "6393578" "6430530" "5774666" "6122436" "6118790" "5881245" "6065050" "6076734" "6164541" "5367621" "5828370" "5835495" "5892535" "5903264" "5935240" "5945986" "5949955" "6044089" "6052148" "6134201" "6215745").pn. ("6215745" "6272190" "6356178" "6404898" "6445738" "6529146" "6396500" "5937164" "5892825" "6078908" "5742730" "5900908"	USPAT	2004/02/25 13:45

41	5	digital adj video adj system$1.asn.	USPAT	2004/02/25 13:47
42	51494	(stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data)) near8 (interleav$3 or combin$3 or synchroniz$8)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:48
43	423	((stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data)) near8 (interleav$3 or combin$3 or synchroniz$8)) and (((725/41) or (725/46) or (725/89) or (725/92) or (725/105) or (725/1) or (725/2) or (725/87)).CCLS.)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:48
44	128	(((725/41) or (725/46) or (725/89) or (725/92) or (725/105) or (725/1) or (725/2) or (725/87)).CCLS.) and (709/231.ccls. or 709/217-219.ccls.)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:48
45	39	((((725/41) or (725/46) or (725/89) or (725/92) or (725/105) or (725/1) or (725/2) or (725/87)).CCLS.) and (709/231.ccls. or 709/217-219.ccls.)) and (((stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data)) near8 (interleav$3 or combin$3 or synchroniz$8)) and (((725/41) or (725/46) or (725/89) or (725/92) or (725/105) or (725/1) or (725/2) or (725/87)).CCLS.))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:49
46	7	(((((725/41) or (725/46) or (725/89) or (725/92) or (725/105) or (725/1) or (725/2) or (725/87)).CCLS.) and (709/231.ccls. or 709/217-219.ccls.)) and (((stream$3 or (real adj video) or (real adj audio) or (real adj media) or media or multimedia or digital or audio or video or mpeg or mbone or (media adj player) or (audio adj player$1)) near8 (information or content or data or information or media or audio or video or metadata or (meta adj data)) near8 (interleav$3 or combin$3 or synchroniz$8)) and (((725/41) or (725/46) or (725/89) or (725/92) or (725/105) or (725/1) or (725/2) or (725/87)).CCLS.))) and @ad<19941130	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:50
47	12116	audio near4 (synchroniz$8 or interleav$3 or combin$3)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:51
48	28	audio near4 (synchroniz$8 or interleav$3 or combin$3) near4 (metadata or (meta adj data))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/02/25 13:53

-	109	(725/105).CCLS.	USPAT	2002/04/19 16:13
-	10	plurality adj (media adj server$1)	USPAT	2002/04/16 16:06
-	10	(media adj servers) and geograph$3	USPAT	2002/04/16 16:01
-	0	(media adj servers) and ((725/105).CCLS.)	USPAT	2002/04/16 16:01
-	123	(media adj servers)	USPAT	2002/04/16 16:12
-	2	locate near4 (media adj server$1)	USPAT	2002/04/16 16:03
-	17	plurality adj3 (media adj server$1)	USPAT	2002/04/16 16:06
-	354	(media adj server$1)	USPAT	2002/04/16 16:12
-	885	(709/219).CCLS.	USPAT	2002/04/16 16:12
-	29	((709/219).CCLS.) and ((media adj server$1))	USPAT	2002/04/16 16:16
-	327	(709/231).CCLS.	USPAT	2002/04/16 16:16
-	26	((709/231).CCLS.) and ((media adj server$1))	USPAT	2002/04/16 16:19
-	17	("4577240" | "5128810" | "5148432" | "5163131" | "5166939" | "5206943" | "5218689" | "5249279" | "5301310" | "5313585" | "5355453" | "5414455" | "5442390" | "5448568" | "5469548" | "5508732" | "5510905").PN.	USPAT	2002/04/16 16:17
-	35	(choos$3 or select$4) near5 ((media adj server$1))	USPAT	2002/04/16 16:21
-	30	(choos$3 or select$4) adj3 (video adj server)	USPAT	2002/04/16 16:25
-	3513	geographic adj location$1	USPAT	2002/04/16 16:32
-	6	(geographic adj location$1) and ((media adj server$1))	USPAT	2002/04/16 16:30
-	4	("4814883" | "5200825" | "5424770" | "5600366").PN.	USPAT	2002/04/16 16:26
-	13	5892535.URPN.	USPAT	2002/04/16 16:27
-	741	server$1 and (geographic adj location$1)	USPAT	2002/04/16 16:30
-	1	(server$1 and (geographic adj location$1)) and (((multi adj media) or media) adj clip$1)	USPAT	2002/04/16 16:32
-	56	(server$1 and (geographic adj location$1)) and ((multi adj media) or media adj clip$1)	USPAT	2002/04/16 16:32
-	27	(((multi adj media) or media) adj clip$1)	USPAT	2002/04/16 16:32
-	339	(different or disparate) adj3 (geographic adj location$1)	USPAT	2002/04/16 16:33
-	123	((different or disparate) adj3 (geographic adj location$1)) and media	USPAT	2002/04/16 16:33
-	3506	725/$.ccls.	USPAT	2002/04/16 16:34
-	7	725/$.ccls. and (((different or disparate) adj3 (geographic adj location$1)) and media)	USPAT	2002/04/16 16:34
-	5	(("5440334") or ("5287249") or ("5289545") or ("5283819") or ("5282028")).PN.	USPAT	2002/04/19 16:14
-	3506	725/$.ccls.	USPAT	2002/04/19 16:14

-	16	(("4999806") or ("4987529") or ("4975691") or ("4963995") or ("4941123") or ("4905094") or ("4899299") or ("4845756") or ("4827256") or ("4658093") or ("4611277") or ("4581484") or ("4506387") or ("4504705") or ("4253157") or ("3882538")).PN.	USPAT	2002/04/19 16:19
-	169	bigelow.inv.	USPAT	2002/04/19 16:19
-	1	bigelow.inv. and modem	USPAT	2002/04/19 16:29
-	299	725/$.ccls. and geograph$3	USPAT	2002/04/19 16:29
-	11	(725/$.ccls. and geograph$3) and (709/219.ccls. or 709/217.ccls.)	USPAT	2002/04/19 16:40
-	18	audio adj servers	USPAT	2002/04/19 16:46
-	289	server$1 same (geographic$4 adj location$1)	USPAT	2002/04/19 16:49
-	0	map same (geographic$3 and (media adj server$1))	USPAT	2002/04/19 16:49
-	13	map same (media adj server$1)	USPAT	2002/04/19 16:50
-	6135	map near4 location$1	USPAT	2002/04/19 16:50
-	742	(map near4 location$1) and server$1	USPAT	2002/04/19 16:50
-	114	(map near4 location$1) same server$1	USPAT	2002/04/19 16:50
-	1984	(audio or video) adj2 demand	USPAT	2002/04/19 16:51
-	443	((audio or video) adj2 demand) and map	USPAT	2002/04/19 16:51
-	141	(((audio or video) adj2 demand) and map) and 725/$.ccls.	USPAT	2002/04/19 16:53
-	12	map adj4 location adj4 server$1	USPAT	2002/04/19 16:54
-	323	server adj location$1	USPAT	2002/04/19 16:56
-	1	("5793980").PN.	USPAT	2002/04/19 16:59
-	29	("3882538" | "4253157" | "4504705" | "4506387" | "4581484" | "4611277" | "4658093" | "4827256" | "4845756" | "4899299" | "4905094" | "4941123" | "4975691" | "4987529" | "4999806" | "5001580" | "5051822" | "5057932" | "5109482" | "5132992" | "5195092" | "5237322" | "5247347" | "5253341" | "5262875" | "5282028" | "5283819" | "5289545" | "5297249").PN.	USPAT	2002/04/19 16:57
-	21	5793980.URPN.	USPAT	2002/04/19 16:58
-	2	glaser.inv. and rinehart	USPAT	2002/04/19 17:05
-	9	("4827256" | "5132992" | "5237322" | "5537409" | "5719786" | "5758085" | "5793980" | "5915094" | "5917835").PN.	USPAT	2002/04/19 16:59
-	28	ADDRESS NEAR3 ((MEDIA OR VIDEO OR MULTIMEDIA) ADJ SERVER$1)	USPAT	2002/04/19 17:19
-	49	PLURALITY ADJ2 ((MEDIA OR VIDEO OR MULTIMEDIA) ADJ SERVER$1)	USPAT	2002/04/19 17:20
-	15	SELECT NEAR3 ((MEDIA OR VIDEO OR MULTIMEDIA) ADJ SERVER$1)	USPAT	2002/04/19 18:03
-	0	DATABASE NEAR3(PLURALITY ADJ2 ((MEDIA OR VIDEO OR MULTIMEDIA) ADJ SERVER$1))	USPAT	2002/04/19 18:07
-	0	(STORAGE ADJ MEDIA)WITH(PLURALITY ADJ2 ((MEDIA OR VIDEO OR MULTIMEDIA) ADJ SERVER$1))	USPAT	2002/04/19 17:21

-	168	(BUFFER OR (STORAGE ADJ MEDIA) OR DATABASE) NEAR7((MEDIA OR VIDEO OR MULTIMEDIA) ADJ SERVER$1)	USPAT	2002/04/19 17:38
-	87	MEDIA ADJ RECEIVER	USPAT	2002/04/19 17:25
-	39	SERVER$1 NEAR5(GEOGRAPHIC ADJ LOCATION$1)	USPAT	2002/04/19 17:31
-	105	SERVER ADJ SELECTION	USPAT	2002/04/19 17:31
-	9	LIST ADJ3((MEDIA OR VIDEO OR MULTIMEDIA) ADJ SERVER$1)	USPAT	2002/04/19 17:39
-	17	LIST ADJ10((MEDIA OR VIDEO OR MULTIMEDIA) ADJ SERVER$1)	USPAT	2002/04/19 17:43
-	25	CLIENT ADJ5 STORE$1 ADJ5 LOCATION	USPAT	2002/04/19 18:09
-	0	MENU NEAR3 ADDRESS NEAR3 ((MEDIA OR VIDEO OR MULTIMEDIA) ADJ SERVER$1)	USPAT	2002/04/19 18:03
-	1	TYPE NEAR3 ADDRESS NEAR3 ((MEDIA OR VIDEO OR MULTIMEDIA) ADJ SERVER$1)	USPAT	2002/04/19 18:04
-	37	725/54	USPAT	2002/04/19 18:06
-	106	DATABASE NEAR10 ((MEDIA OR VIDEO OR MULTIMEDIA) ADJ SERVER$1)	USPAT	2002/04/19 18:12
-	34	(DATABASE NEAR10 ((MEDIA OR VIDEO OR MULTIMEDIA) ADJ SERVER$1)) AND 725/$.ccls.	USPAT	2002/04/19 18:07
-	369	(CLIENT OR WORKSTATION) NEAR5 MENU	USPAT	2002/04/19 18:09
-	0	((CLIENT OR WORKSTATION) NEAR5 MENU) AND (DATABASE NEAR10 ((MEDIA OR VIDEO OR MULTIMEDIA) ADJ SERVER$1))	USPAT	2002/04/19 18:09
-	6	((CLIENT OR WORKSTATION) NEAR5 MENU) AND 725/$.ccls.	USPAT	2002/04/19 18:10
-	794	MENU AND 725/$.ccls.	USPAT	2002/04/19 18:10
-	130	(MENU AND 725/$.ccls.) AND (CLIENT OR WORKSTATION)	USPAT	2002/04/19 18:11
-	1266	((MEDIA OR VIDEO OR MULTIMEDIA) ADJ SERVER$1)	USPAT	2002/04/19 18:12
-	305	(((MEDIA OR VIDEO OR MULTIMEDIA) ADJ SERVER$1)) AND MENU$1	USPAT	2002/04/19 18:12
-	152	((((MEDIA OR VIDEO OR MULTIMEDIA) ADJ SERVER$1)) AND MENU$1) AND 725/$.ccls.	USPAT	2002/04/19 18:13
-	6	(((((MEDIA OR VIDEO OR MULTIMEDIA) ADJ SERVER$1)) AND MENU$1) AND 725/$.ccls.) AND @AD<19920102	USPAT	2002/04/19 18:13
-	25	(((((MEDIA OR VIDEO OR MULTIMEDIA) ADJ SERVER$1)) AND MENU$1) AND 725/$.ccls.) AND @AD<19941130	USPAT	2002/04/19 18:18
-	7602	GEOGRAPHIC$4 NEAR5 LOCATION$1	USPAT	2002/04/19 18:18
-	51	(GEOGRAPHIC$4 NEAR5 LOCATION$1) AND (((MEDIA OR VIDEO OR MULTIMEDIA) ADJ SERVER$1))	USPAT	2002/04/19 18:19
-	6	((GEOGRAPHIC$4 NEAR5 LOCATION$1) AND (((MEDIA OR VIDEO OR MULTIMEDIA) ADJ SERVER$1))) AND @AD<19941130	USPAT	2002/04/19 18:20
-	87	map near5 (geographic adj location$1)	USPAT	2002/04/21 16:24
-	23	(map near5 (geographic adj location$1)) and @ad<19941130	USPAT	2002/04/21 16:22
-	89	map with (server) with location$1	USPAT	2002/04/21 16:23
-	6	(map with (server) with location$1) and @ad<19941130	USPAT	2002/04/21 16:23
-	32	list$1 near5 (geographic adj location$1)	USPAT	2002/04/21 16:29
-	3537	select$4 near5 server	USPAT	2002/04/21 16:43
-	50	plurality adj2 ((multimedia or video or audio or media) adj server$1)	USPAT	2002/04/21 16:44

-	4923	map near3 location$1	USPAT	2002/04/21 16:44
-	0	(map near3 location$1) and (plurality adj2 ((multimedia or video or audio or media) adj server$1))	USPAT	2002/04/21 16:44
-	5	(plurality adj2 ((multimedia or video or audio or media) adj server$1)) and map	USPAT	2002/04/21 16:44
-	1408	(audio or video or media) adj clip$1	USPAT	2002/04/21 16:45
-	390	((audio or video or media) adj clip$1) and map	USPAT	2002/04/21 16:45
-	37	(((audio or video or media) adj clip$1) and map) and @ad<19941130	USPAT	2002/04/22 06:07
-	1	("5132992").PN.	USPAT	2002/04/22 06:07
-	1	(("5132992").PN.) and decod$3	USPAT	2002/04/22 06:13
-	1	(("5132992").PN.) and (processor$1 or cpu)	USPAT	2002/04/22 06:37
-	0	(("5132992").PN.) and quality	USPAT	2002/04/22 06:37
-	1	5682325.pn.	USPAT	2002/04/22 06:37
-	0	5682325.pn. and quality	USPAT	2002/04/22 06:40
-	1	5682325.pn. and updat$3	USPAT	2002/04/22 07:38
-	0	5682325.pn. and (clip$1)	USPAT	2002/04/22 07:27
-	0	(5682325.pn. and (clip$1)) and usage$1	USPAT	2002/04/22 07:27
-	0	(5682325.pn. and (clip$1)) and frequency	USPAT	2002/04/22 07:31
-	0	(("5132992").PN.) and pc	USPAT	2002/04/22 07:31
-	0	5682325.pn. and pc	USPAT	2002/04/22 07:31
-	1	5682325.pn. and computer	USPAT	2002/04/22 07:31
-	1	(("5132992").PN.) and computer$1	USPAT	2002/04/22 07:31
-	1	(("5132992").PN.) and updat$3	USPAT	2002/04/22 07:38
-	8	REALNETWORK$1.ASN.	USPAT	2003/03/04 09:22
-	446	(multimedia or (mult adj media) or media) adj player$1	USPAT	2003/05/29 14:26
-	190	((multimedia or (mult adj media) or media) adj player$1) and @ad<199411012 and server$1	USPAT	2003/05/29 14:26
-	7	(((multimedia or (mult adj media) or media) adj player$1) and @ad<19941101) and server$1	USPAT	2003/05/29 14:26
-	57	((multimedia or (mult adj media) or media) adj player$1) and @ad<19941101	USPAT	2003/05/29 14:32
-	327	player$1 and server$1 and @ad<19941101	USPAT	2003/05/29 14:32
-	3602	load adj balanc$3	USPAT	2003/05/29 14:33
-	0	(load adj balanc$3) and ((((multimedia or (mult adj media) or media) adj player$1) and @ad<19941101) and server$1)	USPAT	2003/05/29 14:33
-	250	(player$1 and server$1 and @ad<19941101) and select$4	USPAT	2003/05/29 14:33
-	129	(player$1 and server$1 and @ad<19941101) and media	USPAT	2003/05/29 14:33
-	124	((player$1 and server$1 and @ad<19941101) and media) and ((player$1 and server$1 and @ad<19941101) and select$4)	USPAT	2003/05/29 14:33
-	5114	(choos$4 or select$4) near5 server$1	USPAT	2003/05/29 14:34

-	14188	(connect$4 or session) near4 server$1	USPAT	2003/05/29 14:35
-	967	709/227.ccls.	USPAT	2003/05/29 14:35
-	1	709/227.ccls. and (player$1 and server$1 and @ad<19941101)	USPAT	2003/05/29 14:35
-	5141	(multiple or plurality or group)adj3 server$1	USPAT	2003/05/29 14:47
-	437	709/231.ccls.	USPAT	2003/05/29 14:36
-	79	((multiple or plurality or group)adj3 server$1) and 709/231.ccls.	USPAT	2003/05/29 14:36
-	2	(((multiple or plurality or group)adj3 server$1) and 709/231.ccls.) and @ad<19941101	USPAT	2003/05/29 14:42
-	3	6018771.URPN.	USPAT	2003/05/29 14:40
-	2	("5079767" | "5276442").PN.	USPAT	2003/05/29 14:40
-	34	((multiple or plurality or group)adj3 server$1) and ((multimedia or (mult adj media) or media) adj player$1)	USPAT	2003/05/29 14:43
-	4	("5890910" | "5947746" | "5953005" | "6069310").PN.	USPAT	2003/05/29 14:44
-	60	(multiple or plurality or group)adj3 ((media or (multi adj media) or multimedia) adj server$1)	USPAT	2003/05/29 15:31
-	2	6279040.URPN.	USPAT	2003/05/29 14:50
-	6	("5572645" | "5603058" | "5625404" | "5656539" | "5671225" | "5790176").PN.	USPAT	2003/05/29 14:50
-	3	6275471.URPN.	USPAT	2003/05/29 14:51
-	4	("5918020" | "6006253" | "6031818" | "6128649").PN.	USPAT	2003/05/29 14:51
-	30	("5091849" | "5150472" | "5241671" | "5276679" | "5305195" | "5305389" | "5317732" | "5347632" | "5351276" | "5442749" | "5446740" | "5491820" | "5493677" | "5499330" | "5528739" | "5530852" | "5537526" | "5539886" | "5544051" | "5557515" | "5572643" | "5577042" | "5621874" | "5623652" | "5625818" | "5649186" | "5659729" | "5717914" | "5778372" | "5802292").PN.	USPAT	2003/05/29 14:53
-	49	("3795771" | "4352124" | "4352200" | "4428078" | "4584603" | "4647980" | "4756528" | "4763360" | "4774514" | "4835604" | "4853555" | "4866515" | "4887152" | "4896209" | "4897714" | "4920432" | "4975696" | "5123015" | "5214505" | "5220419" | "5289272" | "5311302" | "5347632" | "5404567" | "5412376" | "5414455" | "5442390" | "5517508" | "5529265" | "5539448" | "5555466" | "5557541" | "5568180" | "5568484" | "5572442" | "5581270" | "5586937" | "5594924" | "5596647" | "5608447" | "5608448" | "5612730" | "5617331" | "5631693" | "5631694" | "5634012" | "5640297" | "5677905" | "5808660").PN.	USPAT	2003/05/29 14:54
-	7	("5195092" | "5287447" | "5325423" | "5351276" | "5371532" | "5541919" | "5544161").PN.	USPAT	2003/05/29 14:58
-	47	5671225.URPN.	USPAT	2003/05/29 14:59
-	448	709/232.ccls.	USPAT	2003/05/29 15:32
-	1167	709/219.ccls.	USPAT	2003/05/29 15:33

-	249	709/247.ccls.	USPAT	2003/05/29 15:33
-	21793	stream$4 same (audio or mpeg or video or multimedia or media)	USPAT	2003/05/29 15:35
-	22323	stream$4 same (audio or mpeg or video or multimedia or media or clip$1)	USPAT	2003/05/29 15:35
-	43657	(encod$3 or decod$3) same (audio or mpeg or video or multimedia or media or clip$1)	USPAT	2003/05/29 15:35
-	59819	(compress$4 or decompress$4) same (audio or mpeg or video or multimedia or media or clip$1 or stream$3)	USPAT	2003/05/29 15:36
-	638	709/213.ccls. or 709/216.ccls.	USPAT	2003/05/29 15:36
-	6023	(stream$4 same (audio or mpeg or video or multimedia or media)) and (stream$4 same (audio or mpeg or video or multimedia or media or clip$1)) and ((encod$3 or decod$3) same (audio or mpeg or video or multimedia or media or clip$1)) and ((compress$4 or decompress$4) same (audio or mpeg or video or multimedia or media or clip$1 or stream$3))	USPAT	2003/05/29 15:37
-	225	((stream$4 same (audio or mpeg or video or multimedia or media)) and (stream$4 same (audio or mpeg or video or multimedia or media or clip$1)) and ((encod$3 or decod$3) same (audio or mpeg or video or multimedia or media or clip$1)) and ((compress$4 or decompress$4) same (audio or mpeg or video or multimedia or media or clip$1 or stream$3))) and ((multiple or plurality or group)adj3 server$1)	USPAT	2003/05/29 15:37
-	109	((stream$4 same (audio or mpeg or video or multimedia or media)) and (stream$4 same (audio or mpeg or video or multimedia or media or clip$1)) and ((encod$3 or decod$3) same (audio or mpeg or video or multimedia or media or clip$1)) and ((compress$4 or decompress$4) same (audio or mpeg or video or multimedia or media or clip$1 or stream$3))) and 709/231.ccls.	USPAT	2003/05/29 15:37
-	85	((stream$4 same (audio or mpeg or video or multimedia or media)) and (stream$4 same (audio or mpeg or video or multimedia or media or clip$1)) and ((encod$3 or decod$3) same (audio or mpeg or video or multimedia or media or clip$1)) and ((compress$4 or decompress$4) same (audio or mpeg or video or multimedia or media or clip$1 or stream$3))) and 709/219.ccls.	USPAT	2003/05/29 15:37
-	20	((stream$4 same (audio or mpeg or video or multimedia or media)) and (stream$4 same (audio or mpeg or video or multimedia or media or clip$1)) and ((encod$3 or decod$3) same (audio or mpeg or video or multimedia or media or clip$1)) and ((compress$4 or decompress$4) same (audio or mpeg or video or multimedia or media or clip$1 or stream$3))) and 709/232.ccls.	USPAT	2003/05/29 16:02
-	921	711/118.ccls.	USPAT	2003/05/29 16:02

-	3	711/118.ccls. and (((stream$4 same (audio or mpeg or video or multimedia or media)) and (stream$4 same (audio or mpeg or video or multimedia or media or clip$1)) and ((encod$3 or decod$3) same (audio or mpeg or video or multimedia or media or clip$1)) and ((compress$4 or decompress$4) same (audio or mpeg or video or multimedia or media or clip$1 or stream$3))) and ((multiple or plurality or group)adj3 server$1))	USPAT	2003/05/29 16:02
-	103694	(workstation or client or player or (media adj player) or (multimedia adj player) or subscriber)	USPAT	2003/06/02 09:56
-	3979	(multimedia adj servers) or (media adj servers) or ((plurality or multiple or group) adj3 servers) or (video adj servers) or (audio adj servers)	USPAT	2003/06/02 09:58
-	169	725/2.ccls. or 725/4.ccls. or 725/41.ccls.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 10:00
-	99	725/74.ccls.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 10:00
-	67	725/82.ccls.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 10:00
-	291	725/87.ccls.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 10:01
-	36	725/89.ccls.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 10:01
-	151	725/91.ccls.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 10:01
-	58	725/94.ccls.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 10:01
-	14	725/94.ccls. and @ad<19941101	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 10:01
-	5	(725/94.ccls. and @ad<19941101) and ((multimedia adj servers) or (media adj servers) or ((plurality or multiple or group) adj3 servers) or (video adj servers) or (audio adj servers))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 10:02
-	24	((workstation or client or player or (media adj player) or (multimedia adj player) or subscriber)) near5 (select$4 or choos$3) near5 ((multimedia adj servers) or (media adj servers) or ((plurality or multiple or group) adj3 servers) or (video adj servers) or (audio adj servers))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 10:04

-	472	((workstation or client or player or (media adj player) or (multimedia adj player) or subscriber)) near5 ((multimedia adj servers) or (media adj servers) or ((plurality or multiple or group) adj3 servers) or (video adj servers) or (audio adj servers))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 10:04
-	4	(((workstation or client or player or (media adj player) or (multimedia adj player) or subscriber)) near5 ((multimedia adj servers) or (media adj servers) or ((plurality or multiple or group) adj3 servers) or (video adj servers) or (audio adj servers))) and 725/91.ccls.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 10:05
-	0	(((workstation or client or player or (media adj player) or (multimedia adj player) or subscriber)) near5 ((multimedia adj servers) or (media adj servers) or ((plurality or multiple or group) adj3 servers) or (video adj servers) or (audio adj servers))) and 725/89.ccls.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 10:05
-	6	(((workstation or client or player or (media adj player) or (multimedia adj player) or subscriber)) near5 ((multimedia adj servers) or (media adj servers) or ((plurality or multiple or group) adj3 servers) or (video adj servers) or (audio adj servers))) and 725/87.ccls.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 10:06
-	0	(((workstation or client or player or (media adj player) or (multimedia adj player) or subscriber)) near5 ((multimedia adj servers) or (media adj servers) or ((plurality or multiple or group) adj3 servers) or (video adj servers) or (audio adj servers))) and 725/82.ccls.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 10:06
-	0	(((workstation or client or player or (media adj player) or (multimedia adj player) or subscriber)) near5 ((multimedia adj servers) or (media adj servers) or ((plurality or multiple or group) adj3 servers) or (video adj servers) or (audio adj servers))) and 725/74.ccls.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 10:06
-	1	(((workstation or client or player or (media adj player) or (multimedia adj player) or subscriber)) near5 ((multimedia adj servers) or (media adj servers) or ((plurality or multiple or group) adj3 servers) or (video adj servers) or (audio adj servers))) and (725/2.ccls. or 725/4.ccls. or 725/41.ccls.)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 10:06
-	77	(((workstation or client or player or (media adj player) or (multimedia adj player) or subscriber)) same ((multimedia adj servers) or (media adj servers) or ((plurality or multiple or group) adj3 servers) or (video adj servers) or (audio adj servers))) and @ad<19941101	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 10:50
-	10	(("5440334") or ("5297249") or ("5289545") or ("5283819") or ("5282028")).PN.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 11:00

-	777	709/231.ccls.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 11:00
-	602	709/232.ccls.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 11:01
-	0	709/232.ccls. and ((((workstation or client or player or (media adj player) or (multimedia adj player) or subscriber)) same ((multimedia adj servers) or (media adj servers) or ((plurality or multiple or group) adj3 servers) or (video adj servers) or (audio adj servers))) and @ad<19941101)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 11:01
-	1	709/231.ccls. and ((((workstation or client or player or (media adj player) or (multimedia adj player) or subscriber)) same ((multimedia adj servers) or (media adj servers) or ((plurality or multiple or group) adj3 servers) or (video adj servers) or (audio adj servers))) and @ad<19941101)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 11:13
-	25	yurt.inv.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 11:22
-	2	5682325.pn.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 11:37
-	2	5822537.pn.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 11:37
-	1	5822537.pn. and compress$4	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 11:44
-	0	5822537.pn. and ip and tcp	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 11:53
-	1	5822537.pn. and bandwidth	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 11:55
-	2	5132992.pn.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 11:55
-	0	5132992.pn. and bandwidth	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 11:55
-	2	5132992.pn. and compress$4	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2003/06/02 12:31

-	16	(("5822537") or ("5835667") or ("5625404") or ("5612742") or ("5583994") or ("5629732") or ("5561456") or ("5734719")).PN.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM TDB	2003/06/02 12:32

US-PAT-NO: 6604144
DOCUMENT-IDENTIFIER: US 6604144 B1
TITLE: Data format for multimedia object storage, retrieval and transfer

---

Abstract Text - ABTX (1):

In a computer environment, a new storage and delivery data format for multimedia object sets increases performance and improves the user experience by reducing the transactions needed to retrieve a set of n objects from n to 1. The object data is interleaved with data definition entries identifying respective object data into a data format comprising a single stream for storage and/or delivery. The data format eliminates the need for multiple, asynchronous transactions thus reducing latency in the data transfer process. Moreover, the data format allows for optimization of how the object data is prioritized and interleaved to achieve desired performance objectives upon delivery of the multimedia objects.

TITLE - TI (1):

Data format for multimedia object storage, retrieval and transfer

Assignee Name - ASNM (1):

Microsoft Corporation

Brief Summary Text - BSTX (8):

Data transfer speed is basically the result of the bandwidth and latency of a connection. Bandwidth is the speed with which data moves over a connection. Improved bandwidth results by increased carrier capacity and/or increased transmission speed over the carrier. For example, a user can increase the bandwidth of a Web connection over an analog phone line carrier by increasing the modem transmission speed from 14,400 bits per second to 28,800 bits per second. A user may similarly increase the bandwidth of a Web connection by using a digital phone line carrier, such as an Integrated Services Digital Network (ISDN) line, capable of carrying two data channels each at 64,000 bits per second. Latency refers to the delay between the time a request for data is made and the time that the data is received. It results from the logistics of establishing and maintaining a connection over a carrier involving various types of handshakes needed to initiate and complete data transfer. Thus, latency is usually independent of the size of the data being sent over the carrier.

Brief Summary Text - BSTX (9):

Data transfers over the Web conform to the HyperText Transfer Protocol (HTTP). HTTP manages connections between those who request information, clients, and those who provide information, servers. In the HTTP model, a client establishes a connection to a remote server and then issues a request. The server processes the request, returns a response and closes the connection. Importantly, the HTTP model only permits a single transaction per connection. In addition, HTTP uses the Transmission Control Protocol (TCP) as a transport layer. A transport layer is a mechanism for the transfer of data between clients and servers that ensures delivery of in sequence, error-free data with no losses or duplications. TCP establishes connections using a three-part handshake as follows: the client sends a connection request, the server responds and the client acknowledges the response, at which time the client can also send data.

Brief Summary Text - BSTX (10):

Certain design features of HTTP interact poorly with TCP causing problems with data transfer performance. Opening a connection to a server creates a latency of at least two round trips over the network because a client must wait for a server to send its connection response. Similarly, TCP uses a mechanism called slow start to negotiate the data transfer speed over a connection. To prevent overrunning a receiving computer, slow start initiates transfer at a low data transfer rate and progressively increases the data transfer rate as more chunks of data are received successfully. When transferring large amounts of data over a single connection, the impact of slow start and connection latency are minimal. However, when transferring small amounts of data over a large number of connections, the impact of slow start and latency from multiple connections is devastating.

Brief Summary Text - BSTX (11):

In the early days of the Web, cumulative delays from slow start and connection latency were tolerable because early Web pages included a single HTML page with one or two images. However, the complexity of Web pages is increasing. Modern Web page designs include more objects, such as bitmaps, audio, ActiveX controls and Java Applets, and thus HTTP requires numerous transactions to transfer each page over the Web. Thus, the cumulative latency due to TCP handshaking and slow start becomes a substantial performance limitation. Moreover, a HTML page includes the names of referenced objects and a browser cannot request these referenced objects until it has retrieved and interpreted the HTML defining the page. Thus, the latency of retrieving the HTML defining a page can create a cascade of delays in requesting embedded objects. Cascaded round trip delays from opening multiple server connections to retrieve embedded objects and subsequent slow start delays often result in situations where users can do no useful work because the system is waiting instead of transmitting data.

Brief Summary Text - BSTX (12):

Lastly, numerous HTTP transactions severely impact the data transfer

performance of high-speed satellite and cable modem systems. In these systems, the transmission speed from a server to a client, the "downstream" connection, is substantially higher than the transmission speed from client to server, the "upstream" connection. For example, wireless satellite links and cable modems are capable of transmitting data "downstream" at 10,000,000 bits/second rate while transmission of "upstream" browser requests often occurs at a rate of 768,000 bits/second (a "fractional" T1 connection, about 13 times slower). For a Web page having numerous objects, the slow speed "upstream" request connection limits data transfer performance due to the response latency of multiple HTTP requests for these objects.

Brief Summary Text - BSTX (14):

The present invention addresses these performance limitations by improving the efficiency of object retrieval and transfer in multimedia computer environments. In contrast to the multiple transactions currently required to retrieve multiple objects in HTTP, the present invention reduces the number of transactions to retrieve a set of n objects from n to 1, thereby substantially reducing the latency due to slow start and cascaded round trip delays from opening multiple server connections. This increases delivery performance and improves the user experience by substantially reducing the delays in the transfer and rendering of images on the user's display.

Brief Summary Text - BSTX (15):

The present invention includes data for multiple objects in a single data stream in an optimized interleaved format. Object data is interleaved with data definition entries identifying respective object data into a data format comprising a single stream for storage and delivery. The data format eliminates the need for multiple, asynchronous transactions. Moreover, the data format enables one to customize and optimize how the object data is prioritized and interleaved to achieve a desired effect on the viewer upon delivery of the multimedia data.

Brief Summary Text - BSTX (16):

One aspect of the present invention includes a data format for storing, transferring and retrieving, in a computer environment having a storage, a stream of objects comprising object data packets stored within the storage for each object in the stream and at least one data definition entry corresponding to each object in the stream, wherein said at least one data definition entry is interleaved with said object data packets such that a data definition entry corresponding to one of the objects in the stream precedes any object data packets of said one object in the stream.

Brief Summary Text - BSTX (17):

Another aspect of the present invention includes a system for transmitting a page comprising a data repository for a stream representing the page, wherein

said stream comprises object data packets interleaved with data definition entries, a browser assembling said object data packets with reference to said data definition entries so as to display the page represented by said stream and a network communicating said stream between said data repository and said browser.

Brief Summary Text - BSTX (18):

Lastly, another aspect of the present invention includes a system for transmitting a page comprising a data repository for a stream representing the page, wherein said stream comprises object data packets interleaved with data definition entries, a browser assembling said object data packets with reference to said data definition entries so as to display the page represented by said stream and a computer communicating said stream between said data repository and said browser.

Drawing Description Text - DRTX (4):

FIG. 2 is an HTTP Get command for retrieving a file from a server.

Drawing Description Text - DRTX (8):

FIG. 6 illustrates a sequence of browser requests and server responses for a progressively rendered Web page, such as shown in FIG. 1b.

Drawing Description Text - DRTX (9):

FIG. 7 is a block diagram of a serial data stream created from multiple, overlapping object requests serviced by a multitasking server.

Drawing Description Text - DRTX (10):

FIG. 8 is a data diagram showing the structure of a serial data stream prepared in the data format of the present invention.

Drawing Description Text - DRTX (16):

FIG. 14 is a flowchart illustrating the process flow of a browser unpacking a data stream encoded in the data format of the present invention.

Detailed Description Text - DETX (5):

FIG. 1a is an example of a computer environment for practicing the present invention. A client 102 communicates with a server 106 by means of a network 104, such as the World Wide Web. The server 106 may include a gateway to a Wide Area Network (WAN) 108 having a plurality of Local Area Networks (LANs) 109. A browser 103, residing on a client 102, displays a page design for a Jammer home page 100 (FIG. 1b) on the World Wide Web. A user can view this page by entering, or selecting a link to, the URL 101 "http://jammer/jamhome.htm" in a

browser, such as Microsoft Explorer or Netscape Navigator, executing on the user's computer. The URL 101 (FIG. 1b) for the Jammer home page informs the browser to obtain the file jamhome.htm from the jammer server. The file jamhome.htm is the base object defining the Jammer home page. To display this page, a browser 103 hierarchically retrieves objects comprising the page. The browser 103 first retrieves a base object having HTML defining the page. To obtain the base object, the browser 103 first opens a network connection to an HTTP server named jammer. Referring now to FIG. 2, the browser then sends an HTTP Get command 110 to the jammer server for a document named jamhome.htm. In response, the jammer server locates and sends the jamhome.htm document to the client 102 using the HTTP response format shown in FIG. 3. The HTTP response format includes a header 130 followed by object data 140 of the requested document.

Detailed Description Text - DETX (6):

Referring to FIG. 3, the header 130 comprises items designed to provide the browser with information regarding the server, its capabilities, the status of the response and properties of the data returned. The first header item 132 identifies the protocol version and status code. For example, data returned for jamhome.htm is in HTTP version 1.0 format and a "200 OK" return code indicates that the server carried out this request successfully. The second header item 134 notifies the browser of the server type and identifies the data encoding used. For example, the jammer server is Microsoft-PWS95/2.0 compatible and transfers data encoded using Multipurpose Internet Mail Extensions (MIME) version 1.0. The next header items 136 inform the browser of the content data type and the last modification date for the data. For example, jamhome.htm is HTML text and the last modification to the text occurred on Sunday, February 11. Finally, the remaining header item 138 notifies the browser of the amount of data transferred. For example, the jamhome.htm file contains 1463 bytes of data.

Detailed Description Text - DETX (7):

The browser checks the HTTP header 130 to determine if its request was successful and to identify the received document type and the amount of object data The browser then removes the HTTP header 130 and places the object data 140 in a cache for quick access at a later time. The browser then identifies the object data type and begins to process the data. For example, the HTTP header 130 indicates that the object data 140 is of the type HTML. The browser thus invokes its HTML handler to display the text and to parse through the HTML object data 140 to find references to other objects on the page. Referring now to FIG. 4, the HTML base object 141 for the Jammer home page 100 contains references to three embedded image objects stored on the jammer server: a background image 142, an under construction image 144 and an Internet Studio image 146. To assemble and display a page such as the Jammer home page 100 of FIG. 1, the browser must retrieve all objects referenced within the retrieved HTML base object.

Detailed Description Text - DETX (8):

As the browser parses through the HTML base object 141 to locate references to embedded objects, it proceeds through the same steps to retrieve the embedded object data. For each reference to an embedded object, the browser first checks its cache to determine if the object data resides in cache. If so, the browser retrieves the object data from cache. Otherwise, the browser opens a connection to the server having the object data and transmits an HTTP Get command to retrieve the object data. The server in turn responds with an HTTP response header with information regarding the server, its capabilities, the status of the response and properties of the object data returned. Note that the embedded objects 142, 144, 146 referenced in FIG. 4 are of the GIF data type so the browser must invoke its GIF handler to process the object data.

Detailed Description Text - DETX (9):

The browser may also request multiple objects simultaneously. To do this, the browser issues multiple requests to the server, each request requiring a separate connection to the server. The browser then processes the data for each object as it arrives from the server. Using a technique called progressive rendering, the browser draws partial objects on the display as data for each of the multiple objects arrives from the server. The result is that a user views the page as the browser progressively assembles it. This is important since a progressive rendering browser maintains a user's interest despite delays caused by moving large objects over a low-bandwidth connection. Furthermore, the user can interact with those portions of the page already displayed without having to wait for the remaining portions to arrive. FIGS. 5A-5E illustrate a sequence of displays 150 (FIG. 5A), 152 (FIG. 5B), 154 (FIG. 5C), 156 (FIG. 5D), 158 (FIG. 5E) for the Jammer home page 100 of FIG. 1 as it undergoes progressive rendering. To support progressive rendering, the browser interprets HTML commands from the base object as it arrives from the server. Upon locating a reference to an embedded object such as image object 142 of FIG. 4, the browser dispatches a request for its object data. Thus, it is possible to have multiple overlapping object data transfers from the server to the browser while the browser is still parsing and interpreting HTML from the base object defining the page.

Detailed Description Text - DETX (10):

FIG. 6 illustrates a sequence of overlapping transactions needed to progressively render the Jammer home page 100 of FIG. 1. The first transaction 160, a get request, initiates transfer of the base object, jamhome.htm, from the server 169 to the browser 168. In the next transaction 161, the server 169 responds by serially transmitting HTML from the jamhome.htm file. As the browser 168 interprets the HTML, it finds a reference to the background image object 142 and initiates the next transaction 162, a corresponding get request for the watermark.gif file. The server 169 continues to serially transmit HTML from the jamhome.htm file while it locates and prepares the watermark.gif file for transfer. The browser 168 then finds a reference to the under construction image object 144 and initiates another transaction 163, a get request for the underconstruction.gif file. The server 169 continues to serially transmit HTML from the jamhome.htm file while it now locates and prepares both the

watermark.gif and underconstruction.gif files for transfer. The browser 168 now locates a reference to the Internet Studio image object 146 and initiates transaction 164, a get request for the istudio.gif file. The server 169 then transfers object data for watermark.gif 165, underconstruction.gif 166 and istudio.gif 167.

Detailed Description Text - DETX (11):

An HTTP server, such as server 169 of FIG. 6, can receive multiple requests from a single browser, such as browser 168, as well as multiple requests from multiple browsers. The server 169 can handle multiple transactions simultaneously using the multitasking services provided by its operating system. These multitasking services allow a server to create and coordinate the execution of objects, or tasks. The operating system allocates a small time slice for each task to run. Because these time slices are very small, the operating system switches between them rapidly to create the impression that multiple tasks are executing simultaneously. However, in reality, the operating system executes small portions of each task sequentially. Although the operating system executes the instructions for each task in the correct order, the actual instructions executed by the operating system across all of its tasks includes small portions of each task interleaved in an essentially random order. Multitasking facilities differ between operating systems. For example, in the Unix operating system, a server spawns multiple processes (i.e., tasks), each to handle a single transaction. Under Windows 95 and Windows NT, both licensed by Microsoft, a server uses multiple threads, with each thread handling a transaction. For Unix, Windows 95 or Windows NT, each thread or process is responsible for retrieving the requested data from a disk, database, executable code or other process, formatting the data for transmission over the network, encoding the data according to the selected method, and sending it to a client's browser by writing the formatted, encoded data to the network connection opened by the client.

Detailed Description Text - DETX (12):

Referring now to FIG. 7, a multitasking server transfers a serial data stream 179 over the Web for progressive rendering on a client. Each time the browser issues a request to the server to retrieve an object, the server creates a task to handle the transaction. As shown in FIG. 7, a HTML thread 171 retrieves, formats and transmits HTML data 172 while a background image thread 173 retrieves object data for a background image 174, a first image thread 175 retrieves object data for a first image 176 and a second image thread 177 retrieves object data for a second image 178. As each of these threads 171, 173, 175, 177 executes during its time slice, it retrieves, formats and transmits a data fragment to the client. Note that the formatting operation includes insertion of an HTTP header in each of the data fragments resulting from execution of the thread 171, 173, 175, 177. Due to time sliced multitasking, data fragments having small portions of the object data for each of these four objects are interspersed to form the serial data stream 179. The interspersed stream enables a progressively rendering browser to display the data fragments as they arrive. Thus, although a single thread, such as the HTML

thread 171, transmits data packets in the correct order, the order of the data packets in the serial data stream 179 is essentially random. This is a result of multitasking as the initiation and completion of tasks to request data, allocate threads and return data from each thread is also essentially random. Moreover, it is not possible to arrange delivery of data fragments in a selected order in these multitasking systems. Upon receipt of the data stream 179, the browser removes HTTP header information and progressively renders images from the data fragments of the respective object data transferred.

Detailed Description Text - DETX (14):

A preferred embodiment of the present invention is a flexible data format that enables the combination of object data from multiple objects comprising a set of related objects, such as a HTML page having embedded images, into a single data stream. In this manner, a browser can retrieve all of the objects in a single request by receiving and processing the data stream as if multiple requests had been made. This flexible data format is advantageous because, unlike an HTTP extension or other protocol, it does not require changing existing HTTP infrastructure or protocols, such as firewalls and proxies. Likewise, the data format does not require changing how the browser works. Thus, one can implement the data format by publishing data in the Jammer format on the server and unpacking a Jammer data stream using a proxy that understands the Jammer data format, and how to unpack it, on the client.

Detailed Description Text - DETX (15):

Referring now to FIG. 8, a data stream 190 prepared according to an embodiment of the present invention is illustrated. The data stream 190 includes a general purpose Stream Header 180 followed by a series of interleaved records of two types: Data Definition Entries and Data Packets. The Stream Header 180 provides information about how the server packed the data into the data stream 190. This includes the version number used to prepare the data stream 190, as well as optional entries to define the number of objects in the data stream 190 and the total size of the data stream 190. The Stream Header 180 is extensible and includes ASCII text with one header per line, similar in format to an HTTP header.

Detailed Description Text - DETX (16):

As shown in FIG. 9, a data definition entry 192 provides information about a single object that is present in the data stream 190. The data definition entry 192 is the first reference to an object having object data within the data stream 190. The data definition entry 192 includes information about the object, but no object data. The information typically includes an object identifier 194, such as its URL, a stream identifier number (SID) 196 to identify data packets belonging to the stream and other previously described HTTP header information 198, such as language encoding, content length of data and the date of last modification. As shown in FIG. 10, a data packet 200 comprises a SID 202, the number of bytes in the data packet 204 and object data 206. Note that a data packet does not have to include all of the object data

for an object. Thus, in the Jammer data format, the contents of an object may be distributed among a plurality of data packets.

Detailed Description Text - DETX (17):

Referring back to FIG. 8, data stream 190 includes data packets for an HTML base object 183, 186, a background image object 184, a first image object 188 and a second image object 189. A data definition entry for an object must precede the first data packet for that object in the data stream 190. Additionally, the data definition entry for an object must precede any references to that object in the data packets for any other objects in the stream. For example, data definition entries, such as 181, 182, 185, 187 (FIG. 8) for embedded images (142, 144, 146) in the stream must precede references to those embedded images in the HTML base object (FIG. 4). The data definition entries facilitate unpacking of the data stream 190 because they define which objects are present in the data stream 190 and provide information about those objects. In this manner, data definition entries in the data stream 190 serve to notify a browser that unpacks the data stream of all objects present in the data stream. Moreover, the process of interleaving object data and data definition entries into a data stream 190 can occur either prior to or at the time the server sends data to the browser. Thus, for Web pages including a static set of objects, the server may create and store a data stream 190 prior to the time a browser requests the page. Similarly, for Web pages including objects having frequently changing data, such as an object having the results of a database query, or for which an object set is dynamically created at the time of a browser request, the server creates and transmits a data stream 190 at the time of the browser request.

Detailed Description Text - DETX (18):

Furthermore, the data format of the present invention permits optimization of the sequence of object data records. By modifying the sequence of object data records in the data stream 190, it is possible to control how a page is displayed on a client's browser. As discussed previously, there is no particular order to the display of data fragments in current browsers supporting progressive rendering. When a request is made for an image, for example, a prior art browser waits for the data from the server and displays the data as it is received. In contrast, a browser of the present invention knows at the time an object is requested whether it is present in the data stream 190 because data definition entries for all objects present in the data stream 190 precede references to those objects. Thus, when an object is being requested, if a browser of the present invention has already determined from the data definition entries that the object is present in the stream, the browser displays the data packets of the object as they are received instead of initiating a get request for the object from the server. The present invention allows configuration of a data stream to have object data prioritized, so that the author of the page may control the order in which the browser actually receives object data. For example, the author of a page may desire that the browser display a particular advertising image first to ensure that a viewer sees the advertising before anything else. Similarly, the author may desire

that a viewer sees all the embedded images on a page prior to the background image. The Jammer data format permits the author to select the sequence of delivery and display of object data to create a desired effect on the viewer.

Detailed Description Text - DETX (19):

The process of unpacking the data stream 190 shown in FIG. 8 occurs as follows. Upon request for the base object, the server returns the data stream 190. As the browser receives the data stream 190, it interprets the stream header 180, recognizes that the data to follow is in the Jammer data format and then begins to unpack the data stream 190. A first data definition entry 181 follows the stream header 180 and provides the browser with information about the base object. A second data definition entry 182 having information on a background image follows the first data definition entry 181. The browser then reads the first data record 183 and recognizes HTML data. The browser invokes its HTML handler to parse and display data from the first data record 183. A second data record 184 follows in the data stream 190. Using information from the second data definition entry 182, the browser determines that data in the second data record 184 corresponds to the background image and reads this data into a cache 234 (FIG. 13).

Detailed Description Text - DETX (20):

Referring again to FIG. 8, the following record in the data stream 190 is a third data definition entry 185. The third data definition entry 185 provides the browser with information on a first image object for later use. The next record in the data stream 190 is additional HTML data 186 from the base object. In parsing this additional HTML data 186, the HTML handler finds a reference to the background image and issues a request for it. The browser's cache manager receives the request for the background image data and returns the data for the background image it holds in cache for the second data record 184. In addition, the HTML handler finds a reference to the first image object while interpreting the additional HTML data 186. As no object data for the first image object exists in the data stream 190 up to this point, the browser dispatches a request for the first image object. The remainder of the data stream 190 includes a data definition entry for a second image object 187 as well as object data for the first image 188 and the second 189 image. Unpacking of the remaining records in data stream 190 proceeds by continuation of the method described above. Unpacking of the stream 190 terminates upon detection of an end of stream indicator 191.

Detailed Description Text - DETX (22):

The Jammer data format is useful in a variety of computer environments having data storage. In a preferred embodiment, a publisher stores object data in a data repository in various formats including the Jammer data format. As shown in FIG. 11, a publisher 210 includes a stream configurator 212, a template 214, an interleavor 216 and a data definition generator 218. The stream configurator 212 parses through a page stored in a content storage 213 to identify references to objects and their locations within the page. Content

storage 213 includes objects having a variety of object data types, such as HTML, JPEG, WAV, GIF and BMP data. The stream configurator 212 also receives information about the objects, such as the sequence in which the objects comprising the page are to be displayed, and produces a stream configuration template 214. The following pseudo-code illustrates an implementation of a template 214 of the present invention:

Detailed Description Text - DETX (26):

A data definition generator 216 interprets the stream configuration template 214 to create a data definition entry for each object included in the data stream 190. An interleavor 218 also interprets the stream configuration template 214 to produce a data stream 190 by interleaving data definition entries provided by the data definition generator 216 with object data provided by content storage 219. The interleavor 218 uses the display sequence information to determine how to interleave the object data in the data stream 190 so as to ensure that the data definition entry for each object precedes any of its object data packets in the data stream 190. Thus, the interleavor 218 writes object data into the data stream 190 for those objects having a lower sequence number prior to the object data for those objects having a higher sequence number. The following pseudo-code illustrates an implementation of a publisher 210 of the present invention:

Detailed Description Text - DETX (28):

Set countOfBytesWritten to 0 The countOfBytesWritten is the counter of how many bytes of the HTML file have been written to the Jammed File. The reason that we need this, is that we must write a data definition entry for a referenced object before we write the reference to that object in the HTML file. Since each record in the configuration template has the offset in the file of where the reference appears, and the records are in the order that they appeared in the HTML file, as we read each record, we can compare the offset with how much of the parent has been written. In this way, we can optimize when we write the data definition entries to the stream. Note that to support nested objects, that is objects whose ObjParent is greater than 1, we would need to maintain counters of how much of each of the parent objects had been written.

Detailed Description Text - DETX (30):

Add the first record to the activeObjectsList the activeObjectsList is the list of objects for which data definition entries have already been written to the stream, and for which we are currently writing data packets.

Detailed Description Text - DETX (33):

Reference is now principally made to FIGS. 8, 11 and 12 which illustrate the process flow of the publisher 210 (FIG. 11) of the present invention. At state 220, the publisher 210 prepares a stream header 180 to indicate that the data to follow is prepared using the data format of the present invention. The

publisher 210 proceeds to state 222 where it generates a stream configuration template 214 including information regarding objects defined by a page, their locations on the page and their sequence of display. At state 224, the publisher 210 prepares data definition entries 192 (FIG. 9) for each object defined by the page. Note that the publisher 210 may generate the template 214 and prepare the data definition entries 192 (FIG. 9) concurrently. At state 226, the publisher 210 interleaves object data packets 200 (FIG. 10) with data definition entries 192 (FIG. 9) according to the sequence defined by the stream configuration template 214 so as to form a data stream 190. Lastly, the publisher 210 appends an end of stream indicator 191 to indicate the end of the data stream 190.

Detailed Description Text - DETX (34):

FIG. 13 illustrates the structure of a browser 230 of one embodiment of the present invention. In contrast to prior art browsers, the browser 230 of the present invention supports data transfer using the data format of the present invention as well as those of standard TCP and HTTP protocols. The browser 230 includes an unpacker 232 to receive an incoming data stream, such as the data stream 190 (FIG. 8), from a server 106 (FIG. 1a). The unpacker 232 communicates with an unpacked object cache (UOC) 234 to store object data received in the Janmuer data format. In addition, the unpacker 232 may communicate with a browser cache 235 to store received object data that is not encoded in the Jammer data format. The browser cache 235 operates in the conventional manner of prior art browsers, such as Microsoft Internet Explorer and Netscape Navigator, which is well known. The unpacker 232 identifies the data type of an object in the data stream 190 and invokes a data handler 236 to interpret the object data. The data handler 236 is capable of interpreting a wide variety of data types, such as HTML, GIF, JPEG, WAV, ActiveX and Java applets. The data handler 236 in turn passes the interpreted object data to an object renderer 238 to render on a display 239 for viewing.

Detailed Description Text - DETX (35):

In conjunction with FIG. 13, FIG. 14 illustrates the process by which a browser 230 (FIG. 13) unpacks a data stream 190 (FIG. 8) encoded in the data format of the present invention. At state 240, the unpacker 232 of the browser 230 processes a request for an object. For example, when the user points and clicks on a hyperlinked object on a Web page, the browser 230 issues a request, such as a URL, for the object. At state 242, the unpacker 232 may determine if the requested object data is in the Jammer format. If not, then the unpacker 232 issues a request to the server 106 (FIG. 1a) at state 244 to return the object data in the conventional format of the prior art. Otherwise, at state 246, the unpacker 232 checks the UOC 234 to determine if the requested object is present. If the object is not present in the UOC 234, the unpacker 232 proceeds to state 248 to issue a request to the server 106 (FIG. 1a) to return the requested object data in the Jammer format. Otherwise, the UOC 234 returns the requested object. The unpacker 232 proceeds to state 250 to determine if the returned data is a data definition entry 192 (FIG. 9). If a data definition entry 192 (FIG. 9) is found, the unpacker 232 proceeds to state 252 to create

an entry in the UOC 234 for the object corresponding to the data definition entry 192 (FIG. 9). Otherwise, an object data packet 200 (FIG. 10) is encountered and the unpacker 232 proceeds to state 254 where it appends data from the data packet 200 (FIG. 10) to the entry in the UOC 234 corresponding to the object and forwards the data to the browser if it has been previously requested. Lastly, the browser 230 proceeds to state 256 where it checks for the end of stream indicator 191 (FIG. 8). If the indicator is found, the process terminates at state 258. Otherwise, the process continues at state 240.

Detailed Description Text - DETX (36):

Now referring principally to FIG. 13, when the unpacker 232 receives a data stream 190 (FIG. 8), it examines the stream header 180 (FIG. 8) to determine if the data is encoded in the Jammer data format. If the data is encoded in the Jammer data format, the unpacker 232 proceeds to the first data definition entry, which identifies the base object. Since the base object is the object that the browser 230 initially requested, an entry is created for this object in the UOC 234 and data packets for this object will be sent to the browser. When the browser 230 encounters a reference to another object, it makes a request to the unpacker 232 for the object. Because a data definition entry for an object must precede any references to the object in the data stream 190 (FIG. 8), when the unpacker 232 searches the UOC 234, the object has an entry in the UOC 234, which returns the found object to the browser 230 immediately. Otherwise, the unpacker 232 searches the browser cache 235 to locate the requested object. If the unpacker 232 is unable to locate the requested object within the UOC 234 or the browser cache 235, the unpacker 232 requests the object from the server. The following pseudo-code illustrates an implementation of an unpacker 232 of the present invention:

Detailed Description Text - DETX (37):

Function GetObject(URL objName) This funtion is what the Browser calls when it wants to get a particular URL. It is implemented by the Unpacker. It returns the data for the object specified by the URL objName. Check the UOC to see if an entry already exists for this URL. If URL is present in UOC return the object from the UOC else if cacheManager has URL return the object from the cache else contact remote host specified in URL, and request object. if host reports error return error code else if object is not jammer encoded format return object to browser else At this point, we have obtained the object from the server, and it is encoded in the Jammer Encoded Format. Get first record from stream if record.type is not data definition entry end connection to server return error code else create, new entry for object in UOC While GetNextRecordFromStream returns success if record.type is a data definition entry add to UOC if record.type is a data packet find the corresponding entry in the UOC, and append the data to the object. if object is currently being read from the UOC by the browser send this data to browser end if end While end else

Detailed Description Text - DETX (39):

In another embodiment of the present invention, the unpacker resides on a web server proxy in communication with the browser 168 (FIG. 6) and with the server 169 (FIG. 6). In this embodiment, the web server proxy intercepts communications between the browser 168 and the server 169. The web server proxy includes an unpacker and an unpacker object cache. In a similar fashion, the unpacker examines the stream header 180 (FIG. 8) to determine if the data stream 190 (FIG. 8) from the server 169 is encoded in the Jammer data format. If so, the unpacker of the web server proxy unpacks the data stream 190 (FIG. 8) as described above and forwards the unpacked data to the browser 168. In this manner, the data format of the present invention may be used with a conventional browser 168 of the prior art.

Detailed Description Text - DETX (40):

Referring back to FIG. 1a, in another preferred embodiment, the data repository comprises a server 106 linked to the World Wide Web 104 portion of the Internet. The server 106 communicates with storage devices for holding object data. These storage devices may communicate directly or indirectly with the server and may reside separately, on the server computer itself, within other servers communicating with the server or among a plurality of computers linked to the Internet. Storage devices include magnetic disk drives, electronic memory, optical disk drives, magnetic tape and other devices used to stored data. The browser comprises a client 102 having a viewing device and the browser communicates with the server 106 through the World Wide Web 104. Viewing devices for the client 102 include CRT monitors, flat panel displays, projection displays and other display devices. In one sense, a server 106 is any device capable of locating, storing, retrieving and transferring object data and a client 102 is any device capable of requesting, receiving and displaying object data. Servers and clients can be stationary or mobile and may communicate using conventional land lines or wireless technologies, such as satellite, cellular or infrared.

Detailed Description Text - DETX (41):

In yet another preferred embodiment, the data repository comprises a server 106 linked to an intranet 104. The intranet 104 functions in a manner similar to that of the World Wide Web. However, for security reasons, the intranet 104 is not connected to the Internet. The server 106 communicates with storage devices for holding object data. These storage devices may communicate directly or indirectly with the server and may reside separately, on the server itself, within other servers communicating with the server or among a plurality of computers linked to the intranet. Storage devices include magnetic disk drives, electronic memory, optical disk drives, magnetic tape and other devices used to store data. The browser comprises a client 102 having a viewing device and the browser communicates with the server 106 through the intranet 104. Viewing devices for the client 102 include CRT monitors, flat panel displays, projection displays and other display devices. In one sense, a server 106 is any device capable of locating, storing, retrieving and transferring object data and a client 102 is any device capable of requesting, receiving and displaying object data. Servers and clients can be stationary or mobile and may

communicate using conventional land lines or wireless technologies, such as satellite, cellular or infrared.

Detailed Description Text - DETX (42):

In yet another preferred embodiment, the data repository comprises a network server 107 linked to a wide area network 108 or to a local area network 109. The network server 107 communicates with storage devices for holding object data. These storage devices may communicate directly or indirectly with the network server and may reside separately, on the network server itself, within other network servers communicating with the network server or among a plurality of computers linked to the wide or local area network. Storage devices include magnetic disk drives, electronic memory, optical disk drives, magnetic tape and other devices used to store data. The browser comprises a network client 103 having a viewing device and the browser communicates with the network server 107 through the wide area 108 or the local area network 109. Viewing devices for the network client 103 include CRT monitors, flat panel displays, projection displays and other display devices. As is well known in the art, network servers 107 and clients 103 are typically computing devices linked into a network using specialized software for communication and data transfer, such as Microsoft Windows NT or Novell Netware. Network servers 107 and clients 103 can be stationary or mobile and may communicate using conventional land lines or wireless technologies, such as satellite, cellular or infrared.

Detailed Description Text - DETX (45):

The data format of the present invention advantageously overcomes important limitations of available and anticipated alternatives. The present invention substantially reduces the impact of latency on object retrieval by reducing the number of get and retrieve transactions to one. Although planned HTTP improvements enable multiple transactions per connection, the protocol extension does not reduce the number of transactions required to get multiple objects needed to display a multimedia page. Similarly, server push, a technique of enabling a server to interpret the HTML base object to find embedded objects, may reduce latency, but again does not reduce the number of transactions needed to transfer and display the page. Instead, server push merely transfers the burden of multiple object retrieval from the browser to the server. Moreover, server push does not permit an author to optimize the sequence of object data transfers. Another proposed alternative, multi-part MIME, permits assembly of multiple objects into a single data stream. However, in contrast to the present invention, multi-part MIME does not permit interleaving of object data, nor does it permit optimization of how objects and object parts are delivered, in a predetermined order.

Detailed Description Text - DETX (46):

Moreover, the data format of the present invention advantageously reduces server loading by eliminating the need for superfluous transactions. For example, a page having 15 objects currently requires 15 transactions to

transfer. Thus, a server capable of handling 150 transactions/second can only deliver 10 pages/sec. In contrast, the present invention typically results in the transfer of a single larger data stream and this may degrade server performance slightly. Thus, even if the server is only capable of handling 100 transactions/second using the data format of the present invention, it can still deliver 100 pages/sec. Hence, a user practicing the present invention realizes a tenfold improvement over existing techniques.

Claims Text - CLTX (1):

1. A computer-readable medium having stored thereon a data structure, comprising: object data packets stored within the storage for each object in the stream; and at least one data definition entry corresponding to each object in the stream, wherein said at least one data definition entry is interleaved with said object data packets such that a data definition entry corresponding to one of the objects in the stream precedes any object data packets of said one object in the stream.

Claims Text - CLTX (2):

2. The computer-readable medium of claim 1, wherein said data definition entry corresponding to said one object in the stream precedes any reference to said one object from any other object in the stream.

Claims Text - CLTX (3):

3. The computer-readable medium of claim 1, wherein said object data packets are interleaved with said data definition entries so as to display objects in the stream in a selected order.

Claims Text - CLTX (4):

4. The computer-readable medium of claim 1, wherein said data definition entry comprises: an object identifier to identify an object in the stream corresponding to said data definition entry; and a stream identifier to identify object data packets belonging to the stream.

Claims Text - CLTX (5):

5. The computer-readable medium of claim 1, wherein said object data packet comprises: object data; a size indicator indicative of the amount of said object data packet; and a stream identifier number to identify object data packets belonging to the stream.

Claims Text - CLTX (8):

8. The computer-readable medium of claim 5, wherein said object data comprises audio data.

Claims Text - CLTX (9):

9. A system for transmitting a page, comprising: a data repository for a stream representing the page, wherein said stream comprises object data packets interleaved with data definition entries; a browser assembling said object data packets with reference to said data definition entries so as to display the page represented by said stream; and a network communicating said stream between said data repository and said browser.

Claims Text - CLTX (10):

10. The system of claim 9, wherein said data repository comprises a server in communication with a storage device.

Claims Text - CLTX (11):

11. The system of claim 10, wherein said server comprises a computing device.

Claims Text - CLTX (12):

12. The system of claim 10, wherein said storage device comprises at least one storage device in direct communication with said server.

Claims Text - CLTX (20):

20. A system for transmitting a page, comprising: a data repository for a stream representing the page, wherein said stream comprises object data packets interleaved with data definition entries; a browser assembling said object data packets with reference to said data definition entries so as to display the page represented by said stream; and a computer communicating said stream between said data repository and said browser.

Claims Text - CLTX (24):

24. A computer-readable medium having computer-executable instructions for performing the steps comprising: preparing a header for indicating the data structure of a data stream; generating a stream configuration template for indicating the presence of embedded object data packets in the data stream wherein each embedded object data packet corresponds to an embedded object; preparing data definition entries for each embedded object data packet; interleaving embedded object data packets with data definition entries; and appending end of stream indicator for marking the end of the data stream.

Claims Text - CLTX (25):

25. The computer-executable instructions of claim 24, wherein the step of generating a stream configuration template includes determining the number of embedded object data packets, the location of embedded objects, and the order

of display of embedded objects.

Claims Text - CLTX (26):

26. The computer-executable instructions of claim 24, wherein the step of preparing data definition entries includes identifying the embedded object data packet, generating data representative of the size of said embedded object data packet, and generating data representative of the identity of the data stream.

Claims Text - CLTX (34):

34. The data structure of claim 28 wherein the contents of the first data field is data selected from the group consisting of text data, image data and audio data.

Current US Original Classification - CCOR (1):

709/231

US-PAT-NO: 6622171
DOCUMENT-IDENTIFIER: US 6622171 B2
TITLE: Multimedia timeline modification in networked client/server systems

---

Abstract Text - ABTX (1):

Multimedia content is streamed over a network system from a server computer to a client computer. The client allows a user to enter a variable playback speed and varies the speed at which the multimedia content is rendered at the client. Time-scale modification technology is used to maintain the original pitch of any audio content, thereby maintaining its intelligibility.

Assignee Name - ASNM (1):

Microsoft Corporation

Brief Summary Text - BSTX (2):

This invention relates to networked client/server systems and to methods of streaming and rendering multimedia content in such systems.

Brief Summary Text - BSTX (4):

Multimedia streaming--the continuous delivery of synchronized media data like video, audio, text, and animation--is a critical link in the digital multimedia revolution. Today, streaming media is primarily about video and audio, but a richer, broader digital media era is emerging with a profound and growing impact on the Internet and digital broadcasting.

Brief Summary Text - BSTX (5):

Synchronized media means multiple media objects that share a common timeline. Video and audio are examples of synchronized media--each is a separate data stream with its own data structure, but the two data streams are played back in synchronization with each other. Virtually any media type can have a timeline. For example, an image object can change like an animated .gif file: text can change and move, and animation and digital effects happen over time. This concept of synchronizing multiple media types is gaining greater meaning and currency with the emergence of more sophisticated media composition frameworks implied by MPEG-4, Dynamic HTML, and other media playback environments.

Brief Summary Text - BSTX (6):

The term "streaming" is used to indicate that the data representing the various media types is provided over a network to a client computer on a real-time, as-needed basis, rather than being pre-delivered in its entirety before playback. Thus, the client computer renders streaming data as it is received from a network server, rather than waiting for an entire "file" to be delivered.

Brief Summary Text - BSTX (7):

The widespread availability of streaming multimedia enables a variety of informational content that was not previously available over the Internet or other computer networks. Live content is one significant example of such content. Using streaming multimedia, audio, video, or audio/visual coverage of noteworthy events can be broadcast over the Internet as the events unfold. Similarly, television and radio stations can transmit their live content over the Internet.

Brief Summary Text - BSTX (8):

In comparison to text-based or paper-based presentations, streaming multimedia presentations are very effective in certain situations. Audio/visual presentations, for example, are able to capture and convey many subtle factors that are not perceivable from paper-based documents. Even when the content is a spoken presentation, an audio/visual recording captures gestures, facial expressions, and various speech nuances that cannot be discerned from text or even from still photographs.

Brief Summary Text - BSTX (9):

Although streaming multimedia content compares favorably with textual content in most regards, one disadvantage is that it requires significant time for viewing. It cannot be "skimmed" like textual content. Thus, information consumers are forced to choose between the efficiency of the written word and the richness of the multimedia experience.

Brief Summary Text - BSTX (10):

The invention described below addresses this disadvantage of prior art streaming multimedia content, allowing more efficient multimedia perusal of streaming multimedia presentations than has previously been possible.

Brief Summary Text - BSTX (12):

The invention utilizes time-scale modification so that a user can vary the speed of streaming content without destroying its intelligibility. In accordance with the invention, a user selects multimedia content from a menu presented at a network client computer. In addition, the user selects a speed factor, indicating the speed at which the multimedia should be rendered relative to its default speed.

Brief Summary Text - BSTX (13):

In response to these selections by the user, the network client contacts a network server and initiates a composite media stream from the server to the client. As it receives the media stream, the client processes the individual media streams of the composite stream, altering their timelines in accordance with the user's speed designation. Time-scale modification techniques are used to alter audio streams to maintain their original pitch.

Brief Summary Text - BSTX (14):

In alternative embodiments, timelines are altered at the server, before streaming to the client. To accomplish this, the server either alters the timelines dynamically, as specific content is requested, or selects from a number of prestored media streams having pre-altered timelines.

Brief Summary Text - BSTX (15):

The invention includes methods of adapting to limited bandwidth situations by composing or selecting composite streams having differing degrees of quality, and/or by composing or selecting streams with timelines that are altered to closely correspond with whatever speed factor has been chosen. In one embodiment of the invention, certain media streams, such as audio streams, take precedence over other streams such as video streams. In this embodiment of the invention, the audio stream is sent with an unaltered timeline, at a rate sufficient to satisfy the consumption requirements of the client, given the current speed factor. The video is then degraded in quality to reduce its bandwidth, so that it can be streamed in whatever bandwidth is not require by the audio.

Drawing Description Text - DRTX (2):

FIG. 1 is a block diagram of a networked client/server system in accordance with the invention.

Drawing Description Text - DRTX (3):

FIG. 2 is a block diagram of a networked computer that can be used to implement either a server or a client in accordance with the invention.

Drawing Description Text - DRTX (4):

FIG. 3 is a block diagram illustrating communications and rendering of a composite media stream in accordance with the invention.

Drawing Description Text - DRTX (6):

FIG. 5 is a diagrammatic illustration of a graphical user interface window having a time-scale-modification tool for changing a playback speed of

streaming multimedia content according to one implementation.

Drawing Description Text - DRTX (9):

FIG. 8 is a block diagram illustrating one embodiment in which a plurality of timeline-altered media streams are stored at and provided from a server.

Drawing Description Text - DRTX (10):

FIG. 9 is a block diagram illustrating another embodiment in which a plurality of timeline-altered media streams are stored at and provided from a server.

Drawing Description Text - DRTX (11):

FIG. 10 is a block diagram illustrating yet another embodiment in which a plurality of timeline-altered media streams are stored at and provided from a server.

Drawing Description Text - DRTX (12):

FIG. 11 is a block diagram illustrating yet another embodiment in which a plurality of timeline-altered media streams are stored at and provided from a server.

Detailed Description Text - DETX (3):

FIG. 1 shows a client/server network system and environment in accordance with the invention. Generally, the system includes a network server computer 10 and a plurality of network client computers 11. The computers communicate with each other over a data communications network. The communications network in FIG. 1 comprises a public network 12 such as the Internet. The data communications network might also include local-area networks and private wide-area networks.

Detailed Description Text - DETX (4):

Server computer 10 has access to streaming media content in the form of different composite media streams. Some composite media streams might be stored as files in a database or other file storage system 13. Other composite media streams might be supplied to the server on a "live" basis from other data source components through dedicated communications channels or through the Internet itself.

Detailed Description Text - DETX (5):

Streaming Media

Detailed Description Text - DETX (6):

In this discussion, the term "composite media stream" describes synchronized streaming data that represents a segment of multimedia content. The composite media stream has a timeline that establishes the speed at which the content is rendered. The composite media stream can be rendered to produce a plurality of different types of user-perceivable media, including synchronized audio or sound, video graphics or motion pictures, animation, textual content, command script sequences, or other media types that convey time-varying information or content in a way that can be sensed and perceived by a human. A composite media stream comprises a plurality of individual media streams representing the multimedia content. Each of the individual media streams corresponds to and represents a different media type and each of the media streams can be rendered by a network client to produce a user-perceivable presentation using a particular presentation medium. The individual media streams have their own timelines, which are synchronized with each other so that the media streams can be rendered simultaneously for a coordinated multimedia presentation.

Detailed Description Text - DETX (7):

There are various standards for streaming media content and composite media streams. The "Advanced Streaming Format" (ASF) is an example of such a standard, including both accepted versions of the standard and proposed standards for future adoption. ASF specifies the way in which multimedia content is stored, streamed, and presented by the tools, servers, and clients of various multimedia vendors. ASF provides benefits such as local and network playback, extensible media types, component download, scalable media types, prioritization of streams, multiple language support, environment independence, rich inter-stream relationships, and expandability. Further details about ASF are available from Microsoft Corporation of Redmond, Wash.

Detailed Description Text - DETX (8):

Regardless of the streaming format used, an individual data stream contains a sequence of digital data units that are rendered individually, in sequence, to produce an image, sound, or some other stimuli that is perceived by a human to be continuously varying. For example, an audio data stream comprises a sequence of sample values that are converted to a pitch and volume to produce continuously varying sound. A video data stream comprises a sequence of digitally-specified graphics frames that are rendered in sequence to produce a moving picture.

Detailed Description Text - DETX (9):

Typically, the individual data streams of a composite media stream are interleaved in a single sequence of data packets. Various types of data compression might be used within a particular data format to reduce communications bandwidth requirements.

Detailed Description Text - DETX (10):

The sequential data units (such as audio sample values or video frames) are associated with both delivery times and presentation times, relative to an arbitrary start time. The delivery time of a data unit indicates when the data unit should be delivered to a rendering client. The presentation time indicates when the value should be actually rendered. Normally, the delivery time of a data unit precedes the presentation time.

Detailed Description Text - DETX (11):

The presentation times determine the actual speed of playback. For data streams representing actual events or performances, the presentation times correspond to the relative times at which the data samples were actually recorded. The presentation times of the various different individual data streams are consistent with each other so that the streams remain coordinated and synchronized during playback.

Detailed Description Text - DETX (14):

FIG. 2 shows a general example of a computer 20 that can be used as a network node or host in accordance with the invention. Computer 20 is shown as an example of a computer that can perform the functions of either server computer 10 or a client computer 11.

Detailed Description Text - DETX (16):

The system bus 23 may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. The system memory includes read only memory (ROM) 24 and random access memory (RAM) 25. A basic input/output system 26 (BIOS), containing the basic routines that help to transfer information between elements within server computer 20, such as during start-up, is stored in ROM 24. Computer 20 further includes a hard disk drive 27 for reading from and writing to a hard disk, not shown, a magnetic disk drive 28 for reading from or writing to a removable magnetic disk 29, and an optical disk drive 30 for reading from or writing to a removable optical disk 31 such as a CD ROM or other optical media. The hard disk drive 27, magnetic disk drive 28, and optical disk drive 30 are connected to the system bus 23 by an SCSI interface 32 or some other appropriate interface. The drives and their associated computer-readable media provide nonvolatile storage of computer readable instructions, data structures, program modules and other data for computer 20. Although the exemplary environment described herein employs a hard disk, a removable magnetic disk 29 and a removable optical disk 31, it should be appreciated by those skilled in the art that other types of computer readable media which can store data that is accessible by a computer, such as magnetic cassettes, flash memory cards, digital video disks, random access memories (RAMs) read only memories (ROM), and the like, may also be used in the exemplary operating environment.

Detailed Description Text - DETX (18):

Computer 20 operates in a networked environment using logical connections to one or more remote computers, such as a remote computer 49. The remote computer 49 may be another personal computer, a server, a router, a network PC, a peer device or other common network node, and typically includes many or all of the elements described above relative to computer 20, although only a memory storage device 50 has been illustrated in FIG. 2. The logical connections depicted in FIG. 2 include a local area network (LAN) 51 and a wide area network (WAN) 52. Such networking environments are commonplace in offices, enterprise-wide computer networks, intranets, and the Internet. In the described embodiment of the invention, remote computer 49 executes an Internet Web browser program such as the "Internet Explorer" Web browser manufactured and distributed by Microsoft Corporation of Redmond, Wash.

Detailed Description Text - DETX (20):

Generally, the data processors of computer 20 are programmed by means of instructions stored at different times in the various computer-readable storage media of the computer. Programs and operating systems are typically distributed, for example, on floppy disks or CD-ROMs. From there, they are installed or loaded into the secondary memory of a computer. At execution, they are loaded at least partially into the computer's primary electronic memory. The invention described herein includes these and other various types of computer-readable storage media when such media contain instructions or programs for implementing the steps described below in conjunction with a microprocessor or other data processor. The invention also includes the computer itself when programmed according to the methods and techniques described below. Furthermore, certain sub-components of the computer may be programmed to perform the functions and steps described below. The invention includes such sub-components when they are programmed as described. In addition, the invention described herein includes data structures, described below, as embodied on various types of memory media.

Detailed Description Text - DETX (23):

As shown in FIG. 1, a network system in accordance with the invention includes a network server 10 from which a plurality of composite media streams are available. In some cases, the composite media streams are actually stored by server 10. In other cases, server 10 obtains the composite media streams from other network sources or devices.

Detailed Description Text - DETX (24):

The system also includes network clients 11. Generally, the network clients are responsive to user input to select or request identified composite media streams. In response to a request for a composite media stream, server 10 streams the requested composite media stream to the network client in accordance with some known format such as ASF. The client renders the data streams to produce the multimedia content.

Detailed Description Text - DETX (25):

In accordance with the invention, a network client also accepts a speed designation from a human user. The speed designation is preferably a speed factor relative to the original or default playback speed of the selected multimedia stream. For example, a speed factor of 1.2 indicates that the composite media stream is to be rendered at 1.2 times its original or default speed, thereby achieving time compression. A speed factor of 0.8 indicates that the composite media stream is to be rendered at 0.8 times its original or default speed, thereby achieving time expansion.

Detailed Description Text - DETX (26):

In response to the speed designation from the user, the system modifies the timelines of the individual media streams of the composite media stream, while keeping the timelines synchronized with each other and while maintaining the original pitch of any audio produced from audio streams. In one embodiment of the invention, such timeline modification is performed by the network client. In other embodiments of the invention, the timeline modification can be performed at the network server, before the media streams are streamed to the network client.

Detailed Description Text - DETX (27):

Timeline modification changes the timeline of the received data streams in accordance with the user speed designation to achieve either time compression or time expansion. With some types of media, such as video streams, this involves either omitting selected frames or modifying the presentation times of the individual data units or video frames. In other cases, such as with audio streams, the time-modification is more difficult--simply changing the presentation times would alter the pitch of the original audio and make it unintelligible. Accordingly, some type of audio processing technique is used to time-compress or time-expand audio streams, while maintaining the original pitch of the audio--thereby maintaining the intelligibility of the audio.

Detailed Description Text - DETX (32):

More information regarding audio time modification is given in an article that appeared in the March, 1997, issue of "ACM Transactions on Computer-Human Interaction" (Volume 4, Number 1, pages 3-38) (1997). For purposes of this disclosure, it can be assumed that audio time modification involves some combination of changing individual data stream samples, dropping certain samples, and adjusting presentation times of any samples that are actually rendered.

Detailed Description Text - DETX (33):

FIG. 3 illustrates an embodiment of the invention in which timeline modification is performed by network client 11. Network server 10 streams a

composite media stream 101 to network client 11 (although not shown in FIG. 3, other communications also take place bi-directionally between server 10 and client 11, such as control-oriented communications). In addition to communications of the media stream 101, there are various control communications between the server The composite media stream has a plurality of individual media streams as described above. For purposes of discussion, it is assumed in this example that the composite media stream has an audio stream and a video stream.

Detailed Description Text - DETX (34):

Each media stream has a timeline, and the timelines of the individual streams are synchronized with each other so that the streams can be rendered in combination to produce coordinated multimedia content at the network client 11. The original timelines correspond to the original recording or rendition of the multimedia material, so that rendering the streams according to their timelines results in presentation speeds that closely match the speed of the original event or performance. In the case of audio streams, the timelines preserve the original speed and pitch of the original audio content.

Detailed Description Text - DETX (35):

The client computer has a demultiplexer component 104 that receives the composite media stream and that separates out the individual media streams from the composite format in which the data is streamed (such as ASF). This results in an audio stream 105 and a video media stream 106. The individual media streams are sent to and received by respective decoders 108 and 109 that perform in accordance with the particular data format being employed. For example, the decoders might perform data decompression.

Detailed Description Text - DETX (36):

The decoded data streams are then sent to and received by time modification components: an audio timeline modification component 110 and a video timeline modification component 111. These components receive input from a human operator in the form of a speed designation as described above. The timeline modification components change the timelines of the received media streams in accordance with the user speed designation to achieve either linear time compression or linear time expansion. With some types of media, such as video streams, this involves either omitting selected frames or modifying the presentation times of the individual data units or video frames. In other cases, such as with audio streams, some type of audio processing technique as the SOLA technique described above is used to time-compress or time-expand audio streams, while maintaining the original pitch of the audio and to also retain the intelligibility of the audio.

Detailed Description Text - DETX (37):

The timeline modification components 110 and 111 produce individual media

streams that are provided to and received by respective renderers 114 and 115. The rendering components render the streams in accordance with their modified timelines, as the streams continue to be streamed from the network server. In alternative embodiments of the invention, timeline modification components 110 and 111 might be eliminated and their functions performed by decoders 108 and 109.

Detailed Description Text - DETX (38):

Note that the speed designation, provided by the user, dictates the rate at which the network client consumes the composite data stream. Because of this, the client communicates the speed designation to the network server when requesting a particular composite media stream. The server responds by streaming the composite media stream at a rate that depends on or is proportional to the speed designation provided by the user. For example, for a speed factor of 2.0, the client consumes data at twice the normal rate. Accordingly, the server streams the composite media stream at twice its normal rate to meet the demands of the client.

Detailed Description Text - DETX (39):

In the described embodiment, the user is allowed to change the speed designation during rendering of the composite media stream. In some cases, however, it may not be possible to change the playback speed without interrupting the playback momentarily. If this is the case, playback resumes as soon as possible, beginning at a point that shortly precedes the point at which playback was discontinued. Thus, there is some overlap in the presentation--when the presentation resumes, the overlap provides context for the new content that follows.

Detailed Description Text - DETX (40):

FIG. 4 illustrates methodological aspects of the invention. Steps performed at network client 11 are shown on the left-hand side of the figure, while steps performed by network server 10 are shown on the right-hand side of the drawing.

Detailed Description Text - DETX (41):

A step 130 comprises selecting multimedia content from the network server to be rendered at the network client. In most cases, a user performs this selection from a menu of available content or via a URL (uniform resource locator) selection. The content will normally be represented by a single composite media stream. In some cases, different media streams might be available for a particular content segment, varying perhaps in quality and in required bandwidth. Preferably, however, the user is unaware of anything except the simple act of selecting a single topic or composite stream.

Detailed Description Text - DETX (43):

A step 134 comprises requesting the selected content from the server at a speed which will satisfy the requirements of the user's speed designation.

Detailed Description Text - DETX (44):

A step 136, performed by server 10, comprises identifying the particular composite media stream corresponding to the selected content. Step 138 comprises streaming the composite media stream to the client. In this embodiment, the composite media stream has its original timeline, which does not necessarily result in the speed that the user has designated for playback.

Detailed Description Text - DETX (45):

Step 140, performed by the client, comprises receiving the streaming content. Step 142 comprises modifying the timeline of the composite media stream in accordance with the speed designation provided by the user. As described above, this involves modifying the timelines of the individual media streams while maintaining their synchronization and intelligibility. Step 144 comprises rendering the composite media stream in accordance with its modified timeline.

Detailed Description Text - DETX (49):

Media screen 204 is the region of the UI within which the multimedia content is rendered. For video content, the video is displayed on screen 204. For non-visual content, screen 204 displays static or dynamic images representing the content. For audio content, for example, a dynamically changing frequency wave that represents an audio signal is displayed in media screen 204.

Detailed Description Text - DETX (50):

Shuttle controls 206 enable the user to control play of the multimedia content. Shuttle controls 206 include multiple play buttons 212(1), 212(2), and 212(3), a stop button 214, a pause button 216, rewind buttons 218 and 220, and fast forward buttons 222 and 224.

Detailed Description Text - DETX (51):

Play buttons 212(1)-212(3) are associated with different playback speeds of the multimedia content. In this illustration, play button 212(1) corresponds to a normal playback speed (i.e., "x1.0"), play button 212(2) corresponds to a faster playback speed with a speed up factor of 25% (i.e., "x1.25"), and play button 212(3) corresponds to an even faster playback speed with a speed up factor of 50% (i.e., "x1.50"). It is noted, however, that more or less than three buttons may be used (e.g., two, four, five, etc.) and may correspond to speeds both above and below the normalized speed of "x1.0".

Detailed Description Text - DETX (52):

The user can actuate one of the play buttons via a UI actuation mechanism, such as a pointer 226 or by tabbing to the desired play button and hitting the "enter" key. Upon selection of a play button, the multimedia player plays the multimedia content at the playback speed associated with the selected play button. For instance, if the user selects play button 212(2) with a 25% speedup factor, the multimedia player plays the content at a playback speed of 1.25 times the original or default playback speed.

Detailed Description Text - DETX (53):

Once the multimedia content is playing at one speed, the user is free to select a new speed by actuating another of the play buttons 212(1)-212(3). Suppose the user decides to slow the content back to normal speed. The user can actuate the "x1.0" play button 212(1) to return the media content to the normal speed. In response to speed changes, the multimedia player is configured to repeat a portion of the multimedia content at the new speed.

Detailed Description Text - DETX (54):

Content information space 210 lists information pertaining to the multimedia content being rendered on the media screen 204. The content information space includes the show name, author and copyright information, and tracking/timing data.

Detailed Description Text - DETX (55):

FIG. 6 shows another implementation of a graphical user interface window 240 for the multimedia player. Like UI 200 of FIG. 5, UI 240 has command bar 202, media screen 204, shuttle controls 206, volume control 208, and content information space 210. This implementation, however, employs only a single play button 242. Actuation of play button 242 initiates play of the multimedia content.

Detailed Description Text - DETX (56):

UI 240 has a scale mechanism 244 to vary the speed of the content during rendering. The scale mechanism has a range of playback speeds 246, which in this example range from 0.5x to 2.5x the normal speed. Scale mechanism 244 also has a movable slider 248 that is movable over the range 246. The user can position the slider 248 at the desired speed at which the multimedia player is to play the multimedia content.

Detailed Description Text - DETX (58):

Once the multimedia content is playing at one speed, the user is free to select a new speed by moving the slider 248 to a new speed. In response to use manipulation of the scale mechanism, the multimedia player repeats a portion of the multimedia content and begins playing at the new speed.

Detailed Description Text - DETX (59):

FIG. 7 shows a third implementation of a graphical user interface window 260 for the multimedia player. In this implementation, UI 260 has a single play button 262 to initiate playback of the multimedia content. UI 260 also has a menu 264 associated with the play button. In this illustration, menu 264 is a drop-down or pull-down menu that opens beneath the play button in response to actuation of a tab 266 adjacent to the play button. Alternatively, menu 264 may be invoked by placing pointer 226 over play button 262 and right clicking a mouse button.

Detailed Description Text - DETX (60):

Menu 264 lists multiple playback speeds from which a user can select. In the illustrated example, five playback speeds are listed: x0.5, x0.75, x1.0, x1.25, and x1.5. The user can select one of the listed speeds to instruct the multimedia player to play the multimedia content at a desired speed. As noted above, the user can select a new speed after the content has begun playing by invoking the menu and selecting the new speed. In response, the multimedia player repeats a portion of the multimedia content and begins playing at the new speed.

Detailed Description Text - DETX (61):

Server-Based Multimedia Time-Scale Modification

Detailed Description Text - DETX (62):

In various embodiments of the invention, the step of modifying the timeline of the requested multimedia content can be performed in the client as described above, in the server, or in both the client and server. In the network environment, it is desirable to avoid performing any significant timeline modification in the server. Otherwise, the server could quickly become overloaded with requests from multiple clients.

Detailed Description Text - DETX (63):

However, in some cases it may be desirable to store multiple versions of media streams at a server and to select particular versions of the media streams depending on the timeline requirements of the client, as designated by the user. One advantage of this method is that is can require comparatively less communications bandwidth between the server and client.

Detailed Description Text - DETX (64):

As a general example, a server might store a plurality of media streams having timelines modified by different factors. When a client requests a composite media stream, the server selects the version of the media stream whose timeline most closely accords with the speed designation set by the user.

If the timeline does not exactly match the speed designation, the client can perform further timeline modification.

Detailed Description Text - DETX (65):

FIG. 8 illustrates a more specific example. In this embodiment, a server 300 stores multiple media streams 301 corresponding to specific multimedia content 302. The media streams are of different types, such as audio and video. In FIG. 8, audio streams are designated by the letter "A" and video streams are designated by the letter "V". Any combination of a single audio stream and a single video stream can be rendered to produce the multimedia content.

Detailed Description Text - DETX (66):

The various individual data streams have timelines that are modified by different degrees. The speed factors are indicated in FIG. 8. In this embodiment, the audio and corresponding video streams are organized as pairs, each pair forming a composite media stream having a timeline that has been modified by a factor of 0.5, 1.0, or 1.5.

Detailed Description Text - DETX (67):

When a client 303 requests multimedia content from server 300, the client identifies both the content and the speed factor. In response, the server selects the audio and video streams that have timelines most closely approximating the identified speed factor, and combines those individual media streams to form the composite media stream. The resulting composite media stream is then sent to the client. When the timeline is accelerated, this saves bandwidth in comparison to sending an unaltered composite media stream having a higher streaming rate to meet the accelerated consumption demands of the client.

Detailed Description Text - DETX (68):

As a further optimization, the server can store composite media streams having different degrees of timeline modification and different degrees of quality. Generally, a media stream of a lower quality will consume less communications bandwidth than a media stream of a higher quality. Before selecting an appropriate media stream, the server determines the available bandwidth between the server and the client. It then selects a combination of individual media streams that provides the best quality while requiring no more than the available bandwidth.

Detailed Description Text - DETX (69):

When the user changes the playback speed, the client requests a new media stream that most closely corresponds to the requested speed. Playback is resumed in the new stream at the same point (relative to the subject content) at which it was discontinued in the old stream. Thus, the new stream is

initiated at some intermediate point rather than at the beginning. When the streams are linearly altered, it is not difficult to determine the appropriate presentation time in the new stream. Specifically, the point in the new timeline equals oldtime(oldfactor/newfactor), where oldtime is the presentation time in the first media stream at which the speed change is to occur, oldfactor is the playback speed or factor of the old media stream, and newfactor is the playback speed or factor of the new media stream.

Detailed Description Text - DETX (70):

FIG. 9 illustrates a further example utilizing this concept. In this case, a server 310 has stored a single audio stream 311 and multiple video streams 312, all corresponding to a single multimedia segment 313. The video streams different in quality and corresponding required bandwidth: low (lo), intermediate (med), and high (hi). However, the video streams all have a common, unmodified timeline.

Detailed Description Text - DETX (71):

When a client 314 requests the multimedia content from server 310, the server determines or notes both the speed factor designated by the user and the available bandwidth. It then selects the video stream that has best available quality while also requiring no more bandwidth (at the requested speed factor) than the difference between the available bandwidth and the bandwidth consumed by the selected audio stream. Again, this allows the system to compensate for various available bandwidths.

Detailed Description Text - DETX (72):

FIG. 10 shows another example, in which a server 320 has stored multiple audio streams 321 and multiple video streams 322, all corresponding to a single multimedia segment 323. The audio streams differ in the degree by which their timelines have been modified. In this example, there are audio streams having timelines modified by factors of 0.5, 1.0, and 1.5. The video streams differ in quality and corresponding required bandwidth: low (lo), intermediate (med), and high (hi). However, the video streams all have a common, unmodified timeline.

Detailed Description Text - DETX (73):

When a client 324 requests the multimedia content from server 320, the server determines or notes both the speed factor designated by the user and the available bandwidth. It then selects an audio stream that most closely accords with the specified speed factor. It then selects the video stream that has best available quality while also requiring no more bandwidth than the difference between the available bandwidth and the bandwidth consumed by the selected audio stream. Again, this allows the system to compensate for various available bandwidths.

Detailed Description Text - DETX (74):

FIG. 11 illustrates yet another embodiment in which multiple media streams are stored at the server for use depending upon available bandwidth and upon the speed factor designated by the user. In this embodiment, a server 330 stores a single audio stream 332 and a plurality of video streams 333, all corresponding to and representing the same multimedia content 331. The audio stream has an unaltered timeline. However, the video streams have different timelines and also vary by quality and corresponding bandwidth requirements. Specifically, in this example the video streams have timelines modified by factors of 0.5, 1.0, and 2.0. For each speed factor, there is a "low" bandwidth video stream having a relatively low quality, and a "high" bandwidth video stream having a relatively high quality.

Detailed Description Text - DETX (75):

At a normal, unaltered playback rate, the audio stream utilizes a bandwidth of 16 Kbps (kilobits per second). The low bandwidth video streams require a bandwidth of 20 Kbps. The high bandwidth streams require a bandwidth of 40 Kbps.

Detailed Description Text - DETX (76):

Now, suppose that a client requests the multimedia content over a communications channel having a bandwidth of 56 Kbps, at a speed factor of 2.0. At this speed factor, the client consumes audio data at twice the normal rate, which in this case is 32 Kbps. That leaves 24 Kbps of available bandwidth. Accordingly, the server selects the low bandwidth video stream with the timeline modified by a factor of 2.0, and combines it with the audio stream to form a composite media stream for streaming to the client. The total required communications bandwidth is 52 Kbps, which is within the limits of the available bandwidth.

Detailed Description Text - DETX (77):

Although the example give with reference to FIG. 11 is relatively specific, this method of bandwidth utilization can be generalized to include other types of media streams. Each stream is assigned a priority. Audio will generally have a high priority. The high-priority streams are given priority when allocating bandwidth. Thus, in the example above, the audio stream is streamed to the client at its full quality, while the video stream is reduced in quality to fit within the remaining bandwidth.

Detailed Description Text - DETX (78):

Furthermore, a stream such as a video stream can sometimes be timeline-modified dynamically at the server without incurring significant overhead. Accordingly, the server can adjust the timeline and quality of the video stream dynamically to match the available bandwidth, eliminating the need to store multiple video streams at the server. As an example of a situation where this might be easily accomplished, an MPEG (Motion Picture Expert Group) video

stream contains independent frames and several levels of dependent frames. One easy way to reduce bandwidth is to simply drop lower-level dependent frames from the video stream.

Detailed Description Text - DETX (80):

The described methods provide efficient ways to accomplish timeline modification in a networked client/server environment. The invention provides a user with a degree of efficiency and convenience that has previously been unavailable. When using the invention, the user can browse or skim through streaming content, thereby reducing the time required to assimilate such content. Furthermore, the invention provides for efficient use of limited bandwidth, thereby maximizing the practical applications of the invention.

Claims Text - CLTX (1):

1. A method of obtaining and presenting multimedia content, comprising the following steps: accepting a speed designation at a network client from a human user; a network server selecting a composite media stream of a plurality of composite media streams to be rendered at the network client, each of the plurality of composite media streams corresponding to the same multimedia content and having timelines modified by different degrees, the selected composite media stream having a modified timeline most closely matching the speed designation, the selected composite media stream having a plurality of individual media streams representing the multimedia content, at least one of the individual media streams comprising an audio stream representing audio content having an original pitch, the individual media streams having timelines that are synchronized with each other; streaming the composite media stream from the network server to the network client; modifying, at the network client, the timelines of the individual media streams in accordance with the speed designation while keeping the timelines synchronized regardless of the speed designation and while maintaining the original pitch of the audio content; rendering the composite media stream at the network client in accordance with the modified timelines of the individual media streams; and wherein the plurality of composite media streams include multiple composite media streams corresponding to the same multimedia content and having the same degree of timeline modification but different degrees of quality, and wherein the selecting the composite media stream further comprises determining an available bandwidth between the network server and the network client and selecting the composite media stream that provides the best quality while requiring no more than the available bandwidth.

Claims Text - CLTX (3):

3. A computer-readable storage medium containing a program for obtaining and presenting multimedia content, the program having instructions that are executable by a network client to perform steps comprising: making a plurality of forward playback speed designations available for selection; accepting one of the plurality of speed designations from a human user; selecting, independently of the accepting step and based on a negotiation protocol between

a network server and the network client, a composite media stream of a plurality of composite media streams from the network server to be rendered at the network client, each of the plurality of composite media streams corresponding to the same multimedia content and having timelines modified by different degrees, the selected composite media stream having a modified timeline most closely according with the accepted speed designation, the composite media stream having a plurality of individual media streams representing the multimedia content, the individual media streams having timelines that are synchronized with each other; streaming the composite media stream from the network server to the network client; modifying, at the network client, the timelines of the individual media streams in accordance with the speed designation so that the timelines of the plurality of individual media streams match the accepted speed designation while keeping the timelines synchronized regardless of the speed designation; rendering the composite media stream at the network client in accordance with the modified timelines of the individual media streams; and wherein the plurality of composite media streams include multiple composite media streams corresponding to the same multimedia content and having the same degree of timeline modification but different degrees of quality, and wherein the selecting the composite media stream further comprises determining an available bandwidth between the network server and the network client and selecting the composite media stream that provides the best quality while requiring no more than the available bandwidth.

Claims Text - CLTX (5):

5. A computer-readable storage medium as recited in claim 3, wherein the streaming step comprises streaming the composite media stream at different rates depending on the speed designation from the human user.

Claims Text - CLTX (6):

6. A networked client/server system, comprising: a network server; a plurality of network clients that communicate with the network server over a data communications network; a plurality of composite media streams available from the network server, each of the plurality of composite media streams corresponding to the same multimedia content and having timelines modified by different degrees, each composite media stream comprising a plurality of individual media streams that can be rendered by the network clients to produce different types of user-perceivable media, the individual media streams including audio streams that can be rendered by the network clients to produce audio, such audio having an original pitch, the individual media streams having timelines that are synchronized with each other; the network server being configured to select a composite media stream from the plurality of composite media streams having a modified timeline most closely according with a user selected speed designation identified by a network client, the network server further streaming the selected composite media stream to the network client at an appropriate rate determined by the network server based at least in part on the speed designation; the network clients being responsive to user input to modify the timelines of the individual media streams of the selected composite media stream while keeping the timelines synchronized with each other

regardless of the speed designation and while maintaining the original pitch of any audio produced from audio streams; and wherein the plurality of composite media streams include multiple composite media streams corresponding to the same multimedia content and having the same degree of timeline modification but different degrees of quality, and wherein the network sever is further configured to select the composite media stream by determining an available bandwidth between the network server and the network client and selecting the composite media stream that provides the best quality while requiring no more than the available bandwidth.

Claims Text - CLTX (7):

7. A networked client/server system as recited in claim 6, wherein each composite media stream includes at least an audio stream and a video stream.

Claims Text - CLTX (8):

8. A computer-readable storage medium containing a program for streaming multimedia content to a receiving computer, the program having instructions that are executable by a sending computer to perform steps comprising: receiving a user-selected speed designation for playback of the multimedia content at a receiving computer; selecting, from a plurality of composite media streams at the sending computer that correspond to the multimedia content and that have modified timelines modified by different degrees, a composite media stream that most closely matches the user-selected speed designation; then streaming the selected composite media stream from the sending computer to the receiving computer for subsequent timeline modification at the receiving computer so that the timelines of individual streams of the selected composite media stream match the user-selected speed designation; and wherein the plurality of composite media streams include multiple composite media streams corresponding to the multimedia content and having the same degree of timeline modification but different degrees of quality, and wherein the selecting the composite media stream further comprises determining an available bandwidth between the sending computer and the receiving computer and selecting the composite media stream that provides the best quality while requiring no more than the available bandwidth.

Claims Text - CLTX (9):

9. A computer-readable storage medium as recited in claim 8, wherein the composite media stream includes at least an audio stream and a video stream.

Claims Text - CLTX (10):

10. A computer-readable storage medium as recited in claim 8, wherein the composite media stream includes at least an audio stream representing audio content and wherein the modifying step comprises modifying the timeline of the composite media stream while maintaining intelligibility of the audio content.

Claims Text - CLTX (11):

11. A method of streaming particular multimedia content from a sending computer to a receiving computer, the method comprising the following steps: storing multiple media streams at the sending computer corresponding to the multimedia content, said multiple media streams having timelines that are modified by different degrees for forward playback and including streams corresponding to at least first and second media types, wherein a pair of the first and second types of media streams can be rendered to produce the multimedia content; accepting a user-selected speed designation for playback of the multimedia content at a receiving computer; the sending computer selecting those stored media streams of the first and second types that have modified timelines most closely according with the speed designation; streaming the selected media streams from the sending computer to the receiving computer; further modifying the timelines of the selected media streams at the receiving computer in accordance with the speed designation before rendering the selected media streams; and wherein the multiple media streams include multiple media streams corresponding to the multimedia content and having timelines modified by the same degree but different degrees of quality, and wherein the selecting the stored media streams further comprises determining an available bandwidth between the sending computer and the receiving computer and selecting the media streams that provide the best quality while requiring no more than the available bandwidth.

Claims Text - CLTX (12):

12. A method as recited in claim 11, further comprising: checking whether the user-selected speed designation exactly matches a speed designation of the selected media streams; and further modifying the timelines of the selected media streams at the receiving computer only if the user-selected speed designation does not exactly match the speed designation of the selected media streams.

Claims Text - CLTX (13):

13. A method of obtaining and presenting multimedia content, comprising the following steps: selecting multimedia content to be rendered at a network client; accepting a speed designation at the network client from a human user independently of the selecting step; selecting, from a plurality of composite media streams at a network server that correspond to the multimedia content, a composite media stream that has a modified timeline most closely according with the speed designation, wherein the selecting occurs via a negotiation protocol between the network client and the network server based on timeline modification indicated by the speed designation, wherein the plurality of composite media streams include multiple composite media streams corresponding to the multimedia content and having the same degree of timeline modification but different degrees of quality, and wherein the selecting the composite media stream further comprises determining an available bandwidth between the network server and the network client and selecting the composite media stream that provides the best quality while requiring no more than the available bandwidth;

the network server streaming the composite media stream to the network client at an appropriate rate determined by the network server and based at least in part on the speed designation, the composite media stream representing the selected multimedia content and including a plurality of individual streams; the network client further modifying timelines of the plurality of individual streams so that the timelines of the plurality of individual streams match the speed designation; rendering the composite media stream at a speed based on the further modified timelines as it is streamed to produce the multimedia content at the network client; varying the speed of the multimedia content depending on the speed designation from the human user while keeping timelines of the plurality of individual streams synchronized regardless of the speed designation.

Claims Text - CLTX (14):

14. A method as recited in claim 13, wherein the step of varying the speed of the presented multimedia content is performed by receiving different composite media streams from the network server, wherein the different composite media streams have different timelines.

Claims Text - CLTX (15):

15. A method as recited in claim 13, comprising a further step of allowing the user to change the speed designation during rendering of the composite media stream.

Claims Text - CLTX (16):

16. A method as recited in claim 13, wherein the varying step uses time-scale-modification to change the speed of the multimedia content while maintaining its intelligibility.

Claims Text - CLTX (17):

17. A method as recited in claim 13, wherein: the multimedia content includes audio content having an original pitch; and the varying step uses time-scale-modification to change the speed of the multimedia content while maintaining the original pitch of the audio content.

Claims Text - CLTX (18):

18. A method as recited in claim 13 wherein more speed designations are available to the human user than there are composite media streams stored at the network server.

Claims Text - CLTX (21):

21. A method as recited in claim 13, wherein at least one of the composite media streams is modified for playback faster than a normal playback speed, and

wherein at least one of the composite media streams is modified for playback slower than the normal playback speed.

Claims Text - CLTX (22):

22. A method as recited in claim 13, wherein the speed designation is a speed factor relative to an original playback speed of the multimedia content.

Claims Text - CLTX (23):

23. A method as recited in claim 13, further comprising the network server selecting the composite media stream based at least in part on a user preference for a particular quality of one or more of the plurality of individual streams.

Claims Text - CLTX (24):

24. A method of obtaining and presenting multimedia content, comprising: selecting multimedia content to be rendered at a network client; accepting a speed designation at the network client from a human user independently of the selecting of the multimedia content; selecting, by a network server and based at least in part on the speed designation, a composite media stream to be streamed to the network client, wherein the composite media stream represents the selected multimedia content, and wherein the selected composite media stream has a timeline modified by a factor different than the speed designation; streaming the composite media stream from the network server; rendering the composite media stream as it is streamed to produce the multimedia content at the network client; varying the speed of the multimedia content depending on the speed designation from the human user; and wherein multiple composite media streams represent the selected multimedia content and have a same degree of timeline modification but different degrees of quality, and wherein the selecting the composite media stream further comprises determining an available bandwidth between the network server and the network client and selecting the composite media stream that provides the best quality while requiring no more than the available bandwidth.

Claims Text - CLTX (25):

25. A method as recited in claim 24, wherein: the multimedia content includes audio content having an original pitch; and the varying uses time-scale-modification to change the speed of the multimedia content while maintaining the original pitch of the audio content.

Claims Text - CLTX (26):

26. A method of obtaining and presenting multimedia content from an intelligent network server to an intelligent network client, the method comprising: selecting a composite media stream from the network server to be rendered at the network client, the composite media stream having a plurality

of individual media streams representing the multimedia content, at least one of the individual media streams comprising an audio stream representing audio content having an original pitch, the individual media streams having timelines that are synchronized with each other; accepting a speed designation at the network client from a human user independently of the selecting; selecting, by the network server, a composite media stream based on the speed designation; streaming the composite media stream from the network server to the network client; modifying the timelines of the individual media streams in accordance with the speed designation while keeping the timelines synchronized and while maintaining the original pitch of the audio content; rendering the composite media stream at the network client in accordance with the modified timelines of the individual media streams; and wherein the network server maintains a plurality of composite media streams representing the multimedia content and having the same degree of timeline modification but different degrees of quality, and wherein the selecting the composite media stream further comprises determining an available bandwidth between the network server and the network client and selecting the composite media stream that provides the best quality while requiring no more than the available bandwidth.

Claims Text - CLTX (27):

27. A computer-readable storage medium containing a program for streaming multimedia content to a network client, the program having instructions that are executable by a network server to: receive a speed designation for playback of the multimedia content at a network client; compose a composite media stream that represents the multimedia content, the composite media stream having a timeline that is modified in accordance with the speed designation; determine, based at least in part on the speed designation, a rate at which the timeline-modified composite media stream is to be streamed to the network client; stream the timeline-modified composite media stream from the network server to the network client at the determined rate; and wherein multiple composite media streams represent the multimedia content and have the same degree of timeline modification but different degrees of quality, and wherein the instructions that are executable by the network server to compose the composite media stream further cause the network server to determine an available bandwidth between the network server and the network client and select, as streams of the composite media stream, streams that provide the best quality while requiring no more than the available bandwidth.

Claims Text - CLTX (28):

28. A computer-readable storage medium as recited in claim 27, wherein the instructions that are executable by the network server to compose the composite media stream comprise instructions that are executable by the network server to modify the timeline of the composite media stream in accordance with the speed designation prior to streaming the timeline-modified composite media stream from the network server to the network client at the determined rate.

Claims Text - CLTX (29):

29. A method of streaming particular multimedia content from a network server to a network client, the method comprising: storing multiple media streams at the network server corresponding to the multimedia content, said multiple media streams having timelines that are modified by different degrees and including streams corresponding to at least first and second media types, wherein a pair of the first and second types of media streams can be rendered to produce the multimedia content; accepting a speed designation for playback of the multimedia content at a network client; selecting those stored media streams of the first and second types that have modified timelines most closely according with the speed designation; streaming the selected media streams from the network server to the network client at a rate that is based at least in part on the speed designation; and wherein the multiple media streams include multiple media streams corresponding to the multimedia content and having timelines modified by the same degree but different degrees of quality, and wherein the selecting the media streams further comprises determining an available bandwidth between the network server and the network client and selecting the media streams that provide the best quality while requiring no more than the available bandwidth.

Claims Text - CLTX (30):

30. A method of obtaining and presenting multimedia content, the method comprising: selecting multimedia content to be rendered at a network client; accepting a speed designation at the network client from a human user independently of the selecting; a network server selecting, from a plurality of composite media streams at a network server that correspond to the multimedia content, a composite media stream that has a modified timeline most closely according with the speed designation; the network server streaming the composite media stream to the network client at an appropriate rate determined by the network server, the composite media stream representing the selected multimedia content and including a plurality of individual streams; the network client further modifying timelines of the plurality of individual streams so that the timelines of the plurality of individual streams match the speed designation; rendering the composite media stream at a speed based on the further modified timelines as it is streamed to produce the multimedia content at the network client; varying the speed of the multimedia content depending on the speed designation from the human user while keeping timelines of the plurality of individual streams synchronized; and wherein the plurality of composite media streams include multiple composite media streams corresponding to the multimedia content and having timelines modified by the same degree but different degrees of quality, and wherein the selecting the composite media stream further comprises determining an available bandwidth between the network server and the network client and selecting the composite media stream that provides the best quality while requiring no more than the available bandwidth.

Claims Text - CLTX (31):

31. A method as recited in claim 30, further comprising the network server selecting the composite media stream based at least in part on a user preference for a particular quality of one or more of the plurality of

individual streams.

Claims Text - CLTX (32):

32. A method of obtaining and presenting multimedia content, the method comprising: selecting multimedia content to be rendered at a network client; accepting a speed designation at the network client from a human user independently of the selecting; a network server modifying, based at least in part on the speed designation, timelines of a plurality of individual streams of a composite media stream representing the selected multimedia content; the network server streaming the composite media stream to the network client at a rate determined by the network server; the network client further modifying timelines of the plurality of individual streams so that the timelines of the plurality of individual streams match the speed designation; rendering the composite media stream at a speed based on the further modified timelines as it is streamed to produce the multimedia content at the network client; varying the speed of the multimedia content depending on the speed designation from the human user while keeping timelines of the plurality of individual streams synchronized; and wherein multiple composite media streams correspond to the selected multimedia content and have a same degree of timeline modification but different degrees of quality, the network server further selecting one of the multiple composite media streams by determining an available bandwidth between the network server and the network client and selecting the composite media stream that provides the best quality while requiring no more than the available bandwidth.

Current US Original Classification - CCOR (1):

709/231

Current US Cross Reference Classification - CCXR (3):

709/219

Other Reference Publication - OREF (4):

John David N. Dionisio and Alfonso F. Cardenas, "A Unified Data Model for Representing Multimedia, Timeline, and Simulation Data," IEEE Transactions on Knowledge and Data Engineering, vol. 10, No. 5, Sep./Oct. 1998, pp. 746-767.